

TEKTRONIX®

4081

**Graphic System
OPERATOR'S
REFERENCE**

Tektronix, Inc.
P.O. Box 500
Beaverton, Oregon 97077
070-1950-00

First Printing SEPT 1977

Copyright © 1977 Tektronix, Inc.

All Rights Reserved.

All software products including this document, all associated tape cartridges and the programs they contain are the sole property of Tektronix, Inc., and may not be used outside the buyer's organization. The software products may not be copied or reproduced in any form without the express written permission of Tektronix, Inc. All copies and reproductions shall be the property of Tektronix and must bear this copyright notice and ownership statement in its entirety.

PRODUCT 4081 Graphic System

This manual supports the following versions of this product: GOS version 3.0 level 0

MANUAL REVISION STATUS

REV.	DATE	DESCRIPTION
@	9/77	Original Issue

CONTENTS

NOTE

A detailed table of contents precedes each section.

SECTION 1	ABOUT THE 4081 OPERATOR'S REFERENCE
SECTION 2	ABOUT THE 4081 GRAPHIC SYSTEM
SECTION 3	THE STANDARD 4081 GRAPHIC SYSTEM
SECTION 4	EXPANDING THE 4081 GRAPHIC SYSTEM
SECTION 5	A SESSION WITH THE 4081
SECTION 6	DEMONSTRATION PROGRAMS
SECTION 7	SWITCHES, MESSAGES, MODES, LIGHTS, AND KEYS
SECTION 8	FILES
SECTION 9	GOS RESIDENT COMMANDS
SECTION 10	RUNNING A PROGRAM
SECTION 11	GOS UTILITY PROGRAMS
SECTION 12	DATA COMMUNICATIONS IN HOST MODE
SECTION 13	GOS ERROR MESSAGES
SECTION 14	INSTALLATION
SECTION 15	MAINTENANCE
SECTION 16	OPERATING CONSIDERATIONS
APPENDIX A	GOS RESIDENT COMMAND SUMMARY
APPENDIX B	GOS UTILITY PROGRAM SUMMARY
APPENDIX C	GENERAL SYSTEM MESSAGES
APPENDIX D	GOS STANDARD COMMAND AND PROGRAM SYNTAX SYMBOLS
APPENDIX E	AVAILABLE 4081 DOCUMENTATION
APPENDIX F	ASCII CODE CHART
APPENDIX G	GLOSSARY
	INDEX

TABLES

Table	Description	Page
4-1	Required Circuit Card Slots	4-18
6-1	Three-Dimensional Manipulation Sequences.....	6-10
6-2	Function Key Operations.....	6-11
6-3	Refresh Line and Reference Point Manipulation	6-13
6-4	Piston Positioning	6-15
6-5	Motor Speed	6-15
6-6	Valve Timing	6-16
8-1	Device Mnemonics	8-31
8-2	System Devices	8-38
8-3	Default Directory Entries.....	8-45
9-1	GOS Resident Command Syntax Symbols.....	9-4
9-2	Character Size Limits	9-7
10-1	General Program Syntax Symbols	10-4
11-1	GOS Utility Program Syntax Symbols	11-5
11-2	Default Number of Directory Entries.....	11-28
11-3	Formatting Errors.....	11-30
13-1	Command Processor Errors	13-3
13-2	Program Execution Errors.....	13-5
13-3	Input/Output Errors.....	13-9
14-1	Physical Dimensions.....	14-5
14-2	Environmental Specifications	14-8
14-3	Tape Cartridge Environmental Specifications.....	14-9
14-4	Extension Cord Requirements.....	14-11
14-5	Standard Power Cord Conductor Requirements	14-11
15-1	Routine Maintenance Schedule	15-3
15-2	Fuse Protection	15-13
16-1	Peripheral Device Power Requirements	16-4

ILLUSTRATIONS

Fig.	Description	Page
1-1	The 4081 Graphic System with optional 4631 Hard Copy Unit and 4905 Mass Storage Module.	Frontispiece
2-1	The 4081 Graphic System.....	2-3
3-1	The standard 4081 Graphic System.....	3-4
4-1	The expanded 4081 Graphic System.....	4-5
4-2	The six options available for the 4905 Mass Storage Module.....	4-6
4-3	The 4631 Hard Copy Unit.....	4-8
4-4	The 4641 Character Printer.....	4-9
4-5	The 4662 Interactive Digital Plotter.....	4-10
4-6	The Plotter dial settings for operation with the 4081.....	4-12
4-7	The 4953 and 4954 Graphics Tablets.....	4-13
4-8	The Hexadecimal Display Panel.....	4-15
4-9	The PROM Bootstrap Loader.....	4-16
6-1	Tektronix logo.	6-5
6-2	Multiple Tektronix logos.....	6-7
6-3	LANDER lunar landscape and indicators.....	6-8
6-4	Landed lunar module.....	6-9
6-5	MOTOR Display.....	6-14
6-6	Life history of a simple pattern.....	6-18
6-7	Life history of a Cheshire cat pattern.	6-18
7-1	Storage Display Monitor POWER switch.	7-3
7-2	System power bus and circuit breaker.....	7-4
7-3	4081 POWER switch, POWER light, BUSY light, and IPL button	7-5
7-4	MONITOR ERASE, STORAGE HARD COPY, and VIEW buttons.	7-9
7-5	Keyboard prompt lights, Joyswitch, function keys, and JOYSWITCH TERMINATOR key.....	7-14
7-6	Keyboard.....	7-16
7-7	The eight positions of the Joyswitch	7-17

Fig.	Description	Page
8-1	The Cartridge Tape Unit.	8-4
8-2	The Tektronix Tape Cartridge.	8-5
8-3	Inserting a tape cartridge into the Cartridge Tape Unit drive.	8-6
8-4	The BUSY light on the Cartridge Tape Unit drive.	8-7
8-5	Write-protecting the tape cartridge.	8-8
8-6	The 4905 Mass Storage Module option combination 31,33 (with two Flexible Disc Unit drives).	8-9
8-7	The Tektronix Flexible Disc.	8-10
8-8	The green light in the POWER ON switch is lit when the Flexible Disc Unit is on.	8-12
8-9	Opening the door on the Flexible Disc Unit drive.	8-12
8-10	Inserting a flexible disc into the Flexible Disc Unit drive.	8-13
8-11	The write-protect notch in the flexible disc.	8-14
8-12	Covering the write-protect notch with tape.	8-15
8-13	Write-protecting the Flexible Disc Unit drive.	8-16
8-14	The 4905 Mass Storage Module option combination 31,33 (with one Hard Disc Unit).	8-19
8-15	The Tektronix Hard Disc Cartridge.	8-20
8-16	The white light in the POWER switch is lit when the Hard Disc Unit is on.	8-21
8-17	The lower half of the READY/LOAD indicator light must be lit before inserting the hard disc cartridge.	8-21
8-18	Pulling the Hard Disc Unit drive out from the cabinet.	8-22
8-19	Removing the bottom cover from the hard disc cartridge.	8-23
8-20	Placing the hard disc cartridge in the Hard Disc Unit drive.	8-24
8-21	Placing the bottom cover on the inserted hard disc cartridge.	8-25
8-22	The top half of the READY/LOAD indicator light must be lit before attempting to read data from a hard disc.	8-26
8-23	Write-protecting the non-removable hard disc.	8-27
8-24	Write-protecting the hard disc cartridge.	8-28
8-25	The mass storage medium is first divided into equal-sized blocks of 256 bytes each.	8-44
8-26	Blocks that are found defective are labeled bad blocks and are automatically read and write-protected.	8-44
8-27	Directory blocks are reserved at the beginning of the for- matted medium. A header is created in each directory block. ...	8-46
8-28	An entry for the new library file is created in the primary directory.	8-51
8-29	The specified number of blocks are reserved for the library file.	8-51
8-30	Directory blocks are reserved at the beginning of the library file. A header is created in each directory block.	8-53

Fig.	Description	Page
11-1	The SQUISH operation.	11-50
12-1	Data communications between the 4081 and a host computer.....	12-3
12-2	Format for transmitting a character.....	12-4
12-3	Full-duplex data transmission.....	12-7
13-1	Program Status Word (PSW) format	13-7
14-1	4081 Graphic System shipping dimensions.....	14-4
14-2	4081 Graphic System Dimensions.	14-6
14-3	4905 Mass Storage Module Dimensions.....	14-7
14-4	Power cord exit.	14-10
15-1	Cartridge Tape Unit drive.....	15-4
15-2	Tape head location.....	15-5
15-3	Tape head damage.....	15-6
15-4	Cooling vents.....	15-7
15-5	Keyboard prompt lights.....	15-9
15-6	Power supply fuses.	15-10
15-7	Storage Display Monitor fuse.	15-11
15-8	Cartridge Tape Unit fuse.....	15-12
15-9	Tape cartridge respooling procedure.....	15-15
16-1	Circuit breaker and system power bus.....	16-3
16-2	System power bus halves.....	16-5
16-3	Rear panel and cable port.	16-6
16-4	Tape cartridge components.....	16-6
16-5	FOCUS control.....	16-9
16-6	WRITE THRU INTENSITY and HARD COPY INTENSITY controls.	16-10
16-7	Storage of a refresh image.....	16-11

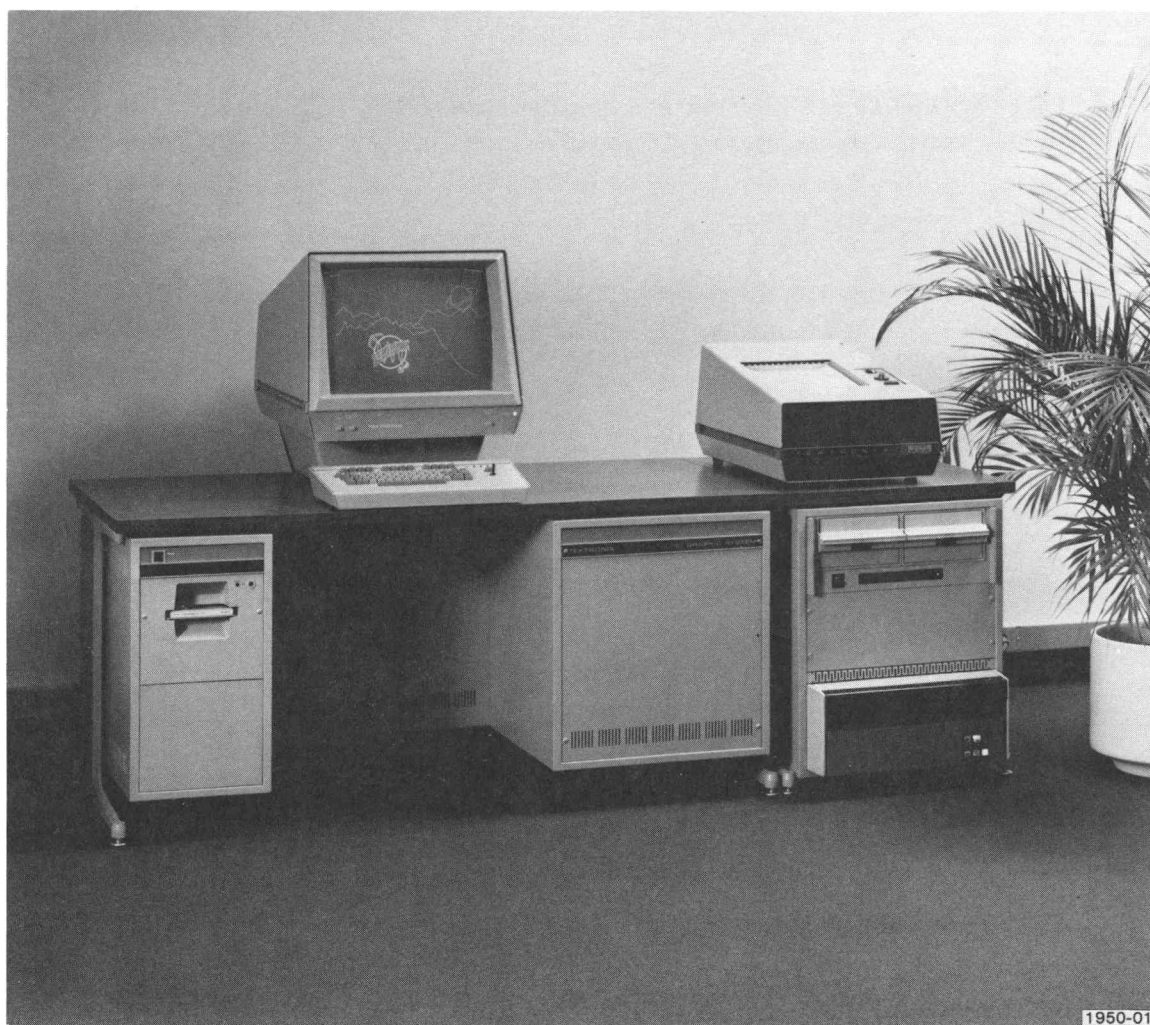


Fig. 1-1. The 4081 Graphic System with optional 4631 Hard Copy Unit and 4905 Mass Storage Module.

Section 1

ABOUT THE 4081 OPERATOR'S REFERENCE

The 4081 Operator's Reference provides information needed to operate the 4081 Graphic System. The manual is written so that people with limited computer experience can learn to run programs and perform basic tasks on the 4081. The 4081 Operator's Reference also provides all 4081 operators, programmers, and technicians an overview of the 4081's components and capabilities.

While reading the manual, the operator may find it helpful to refer to the appendixes in the back of the manual. Most of the computer terms used in the manual are defined in the appendix **Glossary**. The symbols, conventions, and phrases used in the examples of command and program syntax are explained in the appendix **GOS Standard Command and Program Syntax Symbols**. Notice that syntax information enclosed in square brackets ([]) is optional. Do not type the brackets. Also notice in the syntax examples that characters printed in boldface type (for example, the prompt character #) represent information that is actually displayed on the screen by the Graphic Operating System (GOS) or a program running on the 4081. The operator need only type the characters that are *not* printed in boldface type.

For further information on the 4081 Graphic System, individual 4081 components, and Plot 80: software packages, see the appendix **Available 4081 Documentation**.

Note

The 4081 Operator's Reference is valid only for GOS version 3.0 level 0 and subsequent versions of the GOS software.

Section Summary

The 4081 Operator's Reference contains sixteen sections and seven appendixes. A detailed table of contents precedes each section.

Section 1 - ABOUT THE 4081 OPERATOR'S REFERENCE

Briefly describes each section and appendix in the manual.

Section 2 - ABOUT THE 4081 GRAPHIC SYSTEM

Describes computer graphics systems in general and the 4081 Graphic System in particular.

Section 3 - THE STANDARD 4081 GRAPHIC SYSTEM

Describes the individual parts of the standard 4081 Graphic System.

Section 4 - EXPANDING THE 4081 GRAPHIC SYSTEM

Describes the optional devices and programs available for the 4081 Graphic System.

Section 5 - A SESSION WITH THE 4081

Guides new 4081 operators through a first session on the 4081.

Section 6 - DEMONSTRATION PROGRAMS

Describes how to run the demonstration programs that are included as part of the standard 4081 Graphic System.

Section 7 - SWITCHES, MESSAGES, MODES, LIGHTS, AND KEYS

Provides basic 4081 operating procedures. Describes the power switches, the GOS IPL procedure, standard system messages and prompt characters, operating modes, the prompt lights, and keyboard operations.

Section 8 - FILES

Explains the important 4081 file-related concepts. Includes information on file-structured devices, file and device specifiers, formatting a mass storage medium, library files, and file processing.

Section 9 - GOS RESIDENT COMMANDS

Describes the GOS resident commands and how to execute them.

Section 10 - RUNNING A PROGRAM

Describes how to run a program on the 4081. Includes information on Concise Command Language (CCL) and wild cards.

Section 11 - GOS UTILITY PROGRAMS

Describes the GOS utility programs and how to run them.

Section 12 - DATA COMMUNICATIONS IN HOST MODE

Describes the procedure for using the 4081 as a terminal to communicate with a host computer.

Section 13 - GOS ERROR MESSAGES

Describes the GOS standard error messages. Includes probable causes for each error and possible solutions.

Section 14 - INSTALLATION

Describes the customer's responsibilities concerning installation of the 4081 Graphic System. Lists the specifications, dimensions, and requirements to consider when choosing a site for the 4081.

Section 15 - MAINTENANCE

Describes the routine and special 4081 maintenance that the operator can perform.

Section 16 - OPERATING CONSIDERATIONS

Contains additional operator's information to ease 4081 operation and improve system performance. Includes information on the system power bus, tape cartridge care, and the Storage Display Monitor controls.

Appendix A - GOS RESIDENT COMMAND SUMMARY

Provides GOS resident command syntax and brief descriptions.

Appendix B - GOS UTILITY PROGRAM SUMMARY

Provides GOS utility program syntax and brief descriptions.

Appendix C - GENERAL SYSTEM MESSAGES

Lists and explains the common messages and characters displayed on the screen by GOS.

Appendix D - GOS STANDARD COMMAND AND PROGRAM SYNTAX SYMBOLS

Lists and explains the symbols, conventions, and phrases used in the examples of command and program syntax in the manual.

Appendix E - AVAILABLE 4081 DOCUMENTATION

Lists sources for additional information on the 4081, individual 4081 components, and Plot 80: software packages.

Appendix F - ASCII CODE CHART

Provides a chart of the ASCII characters and their codes. The codes are listed in binary, decimal, and hexadecimal notation.

Appendix G - GLOSSARY

Defines most of the 4081 and general computer-related terms used in the manual.

SECTION 2

ABOUT THE 4081 GRAPHIC SYSTEM

CONTENTS

	Page
Computer Graphics	2-4
Computer Graphics Systems	2-4
Input	2-4
Processing	2-5
Output	2-5
The Advantages of the 4081 Graphic System	2-6
A Combined Storage and Refresh Display	2-6
A Graphics System and Intelligent Graphics Terminal	2-7
Ready-Made System Software	2-8
A Variety of System-Compatible Devices	2-9

Section 2

ABOUT THE 4081 GRAPHIC SYSTEM

The 4081 Graphic System (Fig. 2-1) combines the features of low-cost graphics terminals and expensive graphics systems to offer the user a powerful tool for the creation, display, and manipulation of graphics information. The 4081's versatility allows the user to choose how to best implement a particular graphics application.



Fig. 2-1. The 4081 Graphic System.

COMPUTER GRAPHICS

A computer is basically a tool for the processing and storage of information. It is also, like books, radio, and television, a medium for communication. Communication is not only the transmission of information from one person to another. It is a technique for expressing ideas effectively.

Whether explaining an idea to another person or attempting to make an idea clearer to oneself, it is often best to draw a picture. For example, showing a schematic to a technician is a more effective way to describe a circuit board than just listing all the transistors, resistors, capacitors, and integrated circuits that make up the circuit board.

Computer graphics is information in picture form. Graphics makes the computer a more effective communications medium. An aerospace engineer, for example, might use a computer to figure the trajectory of a rocket under variable conditions. The engineer gives the computer the necessary information like rocket weight, thrust, and atmospheric pressure and the computer figures the trajectory. The computer returns the result in terms of such factors as height and distance. On a graphics computer system, however, the engineer can actually view the results. The rocket can be shown moving across the display screen in simulated flight. From pictures of the rocket's flight, the engineer can more easily analyze and compare different trajectories.

COMPUTER GRAPHICS SYSTEMS

All computer graphics systems require three basic components:

- A means to input graphics information into the system.
- A means to process the graphics information.
- A means to output the processed graphics information.

Input

If a computer graphics system is to serve a variety of graphics applications, it should offer the user several ways to input graphics information. A common way is to connect the graphics system to a large host computer and input data from the host computer's picture data base.

This method is limited. The user also needs a way to modify the picture sent from the host computer. The user needs to interact with the processing of the picture. An alphanumeric keyboard allows the user to type text and commands and to activate pre-programmed functions. A joystick allows the user to select specific parts of a picture for modification.

The graphics system should also provide a way to input new pictures. A device like a graphics tablet allows the user to trace a picture, then have the picture converted to digital coordinates that can be processed by the computer.

Finally, the graphics system should allow the user to input graphics information that is stored locally. It is cheaper to transfer some pictures from a mass storage device like disc or tape directly connected to the system than to pay the timesharing costs for transferring pictures from a host computer.

Processing

The ability to process graphics information separates the graphics system from the graphics terminal. A graphics terminal cannot operate independently of a host computer. The terminal requires the processing power of the host computer for all graphics applications.

Many graphics applications, however, do not always need the powerful processor offered by a host computer. In addition, host computer communications are often slow and expensive.

A graphics system saves time and cost by doing its own processing of graphics information. A graphics system can operate without the aid of a host computer for most applications. When an application does require more processing power and a larger picture data base, the graphics system should have the ability to communicate with a host computer.

A resident processor also makes it possible to write and execute programs on the graphics system. An operating system program can be written to perform the many routine and often tedious tasks necessary to coordinate the graphic system's functions. An operating system frees the user to concentrate on the graphics application, not the machine. Assemblers, compilers, text editors, and program debugging aids can be written to allow the user to write, edit, compile, debug and run programs on the graphics system, again saving costly time on a host computer.

Output

The basic output device for most, if not all, graphics systems is the display monitor. There are usually two kinds of display monitors available: a storage display and a refresh display.

On a storage display, a picture is drawn on the screen once. The picture remains visible on the screen until the screen is erased. On a refresh display, a picture is not "stored" on the screen. The picture must be continually drawn and redrawn on the screen, generally at a rate of 30 to 60 times a second.

The advantages of a storage display are the clarity of the displayed image, the complexity and size of the picture that can be displayed, and the low cost of a storage display system. Pictures displayed on a storage display, however, are static. Since a stored picture is drawn only once, the picture cannot be modified unless the screen is erased and the modified picture is redrawn on the screen.

On a refresh display, the user can modify a picture and view the results immediately. Since the displayed picture is redrawn on the screen many times a second, changes in the picture appear smooth and fluid. The effect is much like that on a television screen. What the viewer sees on television as a continuous, natural movement is actually thousands of ever-changing images broadcast in rapid succession.

Almost any graphics application that requires modification or movement benefits from a refresh display. The user, for example, can rotate, enlarge, shrink, move, and delete parts of a picture on a refresh display, then view the results immediately. Or, the user can take part in computer animated simulation, like attempting to land a lunar module on the moon.

The refresh display, however, has its disadvantages. To prevent annoying flicker, the size and complexity of the displayed picture is limited. Also, because of the electronics and resident memory required, a refresh display is much more expensive than a storage display.

A display monitor, whether storage or refresh, often is not alone sufficient for the output of graphics information. Pictures and text can be viewed only as long as they are displayed on the screen. The graphics system should provide a way to make lasting copies of the screen image.

There are several devices that allow the user to transfer information from the computer to a piece of paper. A line printer is commonly used for typewriter-like text reproduction. A hard copy unit makes a direct copy of an image displayed on the screen. A plotter provides high-quality drawings of both pictures and text.

THE ADVANTAGES OF THE 4081 GRAPHIC SYSTEM

A Combined Storage and Refresh Display

The 4081 provides a feature that allows pictures and text to be drawn and redrawn on the Storage Display Monitor without actually storing them on the screen. The result is a storage display with refresh display capabilities.

The 4081 user benefits from the advantages of both storage and refresh displays. The low cost of a storage display allows the 4081 to be priced much lower than most refresh-only display systems. A storage display also allows pictures of a greater clarity and complexity than refresh.

Graphics applications that require the modification or movement of pictures are better served by the 4081's refresh capabilities. In many of these applications, however, only a part of the picture is modified or moving. The rest of the picture generally remains unchanged and fixed. On the 4081, the fixed part of the picture can be displayed in storage while the part of the picture to be modified or moved is displayed in refresh.

To illustrate the benefits of combining storage and refresh, consider a sample graphics application in which the course of an orbiting satellite is plotted across a section of the globe. The background map representing a section of the Earth remains fixed as the satellite is moved across the screen. The map shows continents, mountains, rivers, countries, and major cities in great detail. Because of the map's complexity and the clarity of detail required, the map is best displayed in storage. The satellite, however, is best displayed in refresh to show its smooth, continuous movement across the screen.

Many of the demonstration programs that come with the standard 4081 Graphic System make use of its combined refresh and storage capabilities. In the program MOTOR, the motor's moving valves and piston are displayed in refresh, while the fixed outline of the cylinder is displayed in storage. In the program LANDER, the lunar module is displayed in refresh, while the moon's surface is displayed in storage.

It also is possible on the 4081 to display a picture in refresh, modify it, then redisplay the modified picture in storage. An application, for example, might require the positioning of furniture in an office. The furniture is first displayed in refresh. The 4081 operator, using the Joystick on the keyboard, moves the furniture around the room. When the operator is satisfied with the location of the furniture, each piece can be redisplayed in storage. A hard copy can then be made of the final layout.

The 4081 demonstration program XLOGO illustrates this feature. The operator can move the Tektronix logo, displayed in refresh, across the screen, then redisplay the logo in storage, fixing it in the desired location.

A Graphics System and Intelligent Graphics Terminal

The 4081 Graphic System contains two processors:

- A general-purpose minicomputer that is the 4081's central processing unit (CPU).
- A dedicated display microprocessor that processes graphics information for display on the Storage Display Monitor.

The combined processing power enables the 4081 to operate without the aid of a host computer for a variety of graphics applications.

Some applications, however, require the resources of a large computer system. For these applications, the two processors allow the 4081 to communicate with a host computer as an intelligent graphics terminal. An intelligent graphics terminal means that the 4081 does not have to depend on the host computer for all its processing needs.

While the 4081 is connected to a host computer, the 4081's CPU can be executing a program. At the same time, the display processor can be processing picture information for display on the Storage Display Monitor. By making efficient use of the 4081's intelligence, the user saves costly host processing time.

The Interactive Graphics Terminal (IGT) program that comes with the standard 4081 Graphic System employs the 4081's processors for host computer communications. The IGT program running on the 4081 is under the control of the user's application program running on the host computer. By calling host resident subroutines supplied with the IGT program, the user's program initiates the execution of graphics-related operations by the IGT program. In this way, much of the graphics processing required by the application program is performed by the 4081.

Ready-Made System Software

System software consists of programs that simplify the operation and programming of a computer system. The standard 4081 Graphic System comes with its own set of system software, ready to run as soon as the 4081 is installed. The 4081 software allows the user to concentrate on graphics applications immediately instead of spending months designing operating and programming aids.

The standard 4081 system software includes:

- The Graphic Operating System(GOS)
- GOS Utility Programs
- The TEKTRONIX 4014 Computer Display Terminal Emulator
- The Interactive Graphics Terminal (IGT) package
- The 4081 Verification Program

In addition to the standard system software, a set of programming aids is available as an option. The user can order all the software necessary to write, edit, assemble, debug, and run programs on the 4081. For applications that require a higher-level programming language, a graphics-enhanced FORTRAN IV Compiler is also available as an option.

For descriptions of the standard system software, see the section **The Standard 4081 Graphic System**. For descriptions of the optional software available, see the section **Expanding the 4081 Graphic System**.

A Variety of System-Compatible Devices

The standard 4081 Graphic System may not be sufficient for all graphics applications. The user, however, is not limited to the standard system. A wide variety of optional devices compatible with the 4081 are available to handle most graphics applications.

The 4081's Graphic Operating System (GOS) already contains the routines necessary to control additional devices connected to the 4081. Generally, adding a new device to the 4081 requires only inserting the device's controller board in the 4081 cabinet, connecting the device to the controller board, then plugging the device into the 4081 system power bus.

The 4081's device independence allows the user to easily tailor existing application programs to accommodate a new device. By changing one step, for example, a program that displays a picture on the Display Monitor can instead draw the same picture on a plotter or store the graphics information for the picture on a flexible disc.

Standard GOS utility programs like PLOT and PRINT allow the user to initiate common operations for new devices simply by typing a few characters on the 4081 keyboard.

The available optional devices include:

- The 4905 Mass Storage Module (with Flexible Disc and/or Hard Disc Units)
- The 4631 Hard Copy Unit
- The 4641 Character Printer
- The 4662 Interactive Digital Plotter
- The 4953/4954 Graphics Tablets
- The Hexadecimal Display Panel

For descriptions of all available options for the 4081, see the section **Expanding the 4081 Graphic System**.

SECTION 3

THE STANDARD 4081 GRAPHIC SYSTEM

CONTENTS

	Page
The Hardware	3-3
The Central Processing Unit (CPU).....	3-3
The Memory	3-5
The Display Processor	3-6
The Storage Display Monitor.....	3-6
The Keyboard Unit	3-6
The Cartridge Tape Unit.....	3-7
The Communications Interface	3-8
The Software	3-8
The Graphic Operating System (GOS).....	3-9
GOS Utility Programs	3-9
The 4014 Emulator.....	3-10
The Interactive Graphics Terminal (IGT).....	3-10
The 4081 Verification Program.....	3-11
Digitizer	3-12
Demonstration Programs	3-13

Section 3

THE STANDARD 4081 GRAPHIC SYSTEM

The 4081 Graphic System consists of a number of components joined together to provide a means for the input, processing, and output of graphics information. Though the 4081 is treated as a system throughout this manual, it might be helpful to briefly describe the functions of its individual parts.

The parts of a computer system are generally divided into two basic categories: the hardware and the software.

THE HARDWARE

The term hardware refers to the physical components of a computer system. The standard 4081 Graphic System hardware includes:

- A central processing unit (CPU)
- 32K bytes of memory
- A display processor
- A 19-inch Storage Display Monitor
- A Keyboard Unit that includes an alphanumeric keyboard, 12 programmable function keys, four prompt lights, and a Joyswitch
- A Cartridge Tape Unit
- A communications interface

The Central Processing Unit (CPU)

The CPU can be viewed as the nucleus of the 4081 Graphic System (see Figure 3-1). It processes information and directs the operation of all other 4081 devices.

In relation to the CPU, all other 4081 devices play supporting roles. Their basic purpose is to either send information to or receive information from the CPU.

THE STANDARD 4081 GRAPHIC SYSTEM

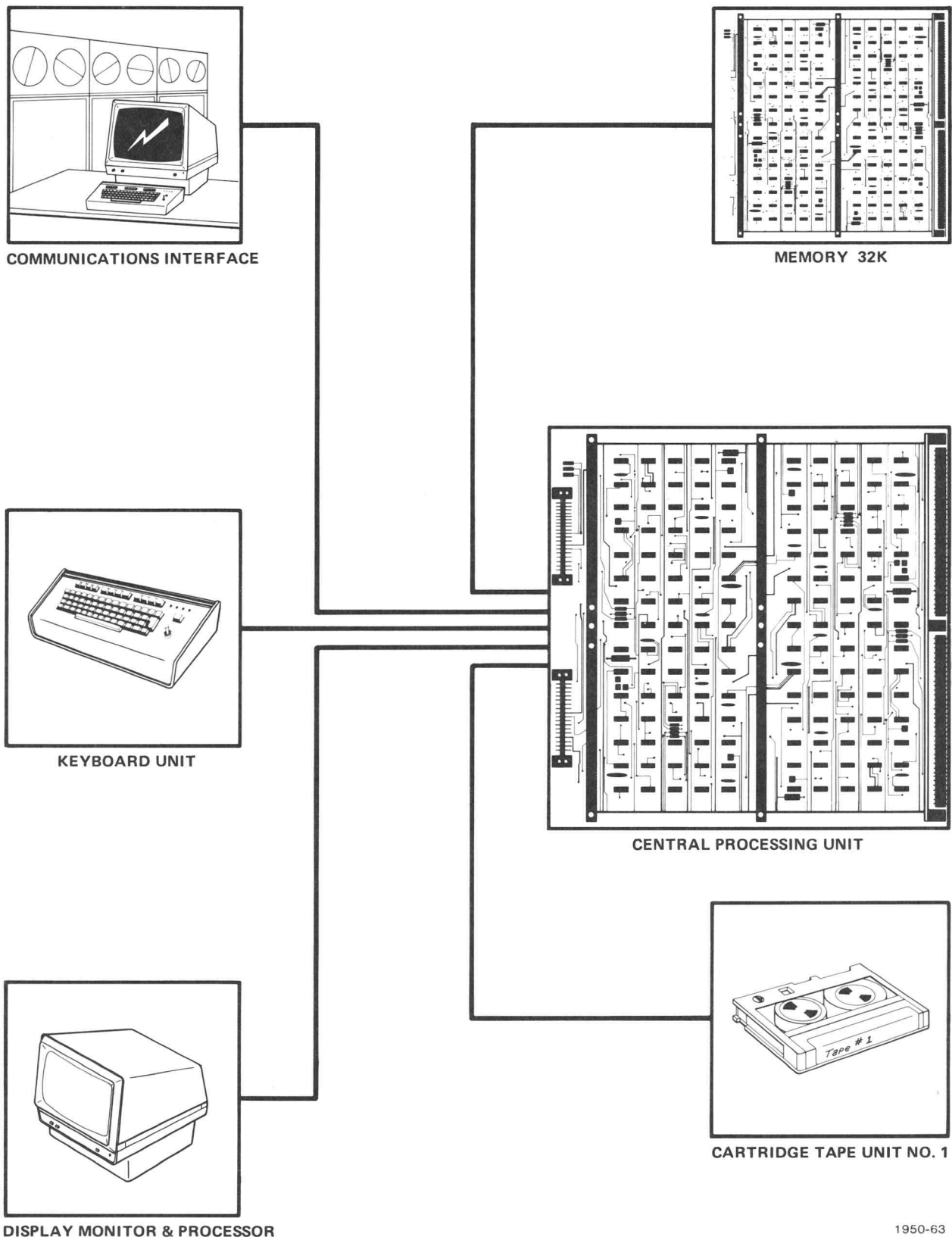


Fig. 3-1. The standard 4081 Graphic System.

The CPU is guided by a set of instructions. A sequence of these instructions makes up a program. The CPU receives and executes the program instructions one by one. Some instructions require the processing of information. Processing information is essentially performing arithmetic and logic operations like addition, subtraction, multiplication, and division on a set of binary numbers. Other instructions require the transfer of information to and from system devices. In this case, the CPU must communicate with the other devices to determine if a device is ready to send information to the CPU or if a device is ready to receive it.

The Memory

The 4081 memory is a device for the storage of information and program instructions. A program must be loaded into memory before the CPU can execute it. The CPU then receives the instructions from memory one at a time. After executing an instruction, the CPU sends for the next one. The transfer of information between memory and the CPU is extremely fast.

The Graphic Operating System (GOS) program is generally always resident in memory when the 4081 is running. Memory also contains utility and user programs in the process of execution (running programs), system information (for example, the current date and time), and refresh graphics information (since refresh pictures are continuously drawn and redrawn on the screen, the picture information must remain resident in memory).

The smallest unit of the 4081 memory is called a binary digit (or bit). A bit has a value of either zero or one. All information in memory is stored as a combination of ones and zeroes.

The 4081 uses ASCII code to store information that is not readily represented by ones and zeroes. This includes most information other than numbers and CPU instructions. ASCII code assigns a binary value (made up of ones and zeroes) to alphanumeric characters, symbols, and control characters. For example, the letter **A** is stored in memory as the binary value **1000001**. (ASCII is an acronym for American Standard Code for Information Interchange. For a complete list of ASCII characters and their assigned binary values, see the appendix **ASCII Code Chart**.)

The 4081 memory consists of semiconductors. Semiconductor memory is considered "volatile". In other words, semiconductor memory requires a consistent source of electrical power in order to retain stored information. For this reason, the GOS program has to be loaded into memory each time the 4081 is turned on.

The standard 4081 memory has a capacity of 32K bytes of data storage; 32K bytes is equal to 32,768 bytes (1K is equal to 1024). A byte is a group of eight consecutive bits.

The Display Processor

The display processor, though under the control of the CPU, does much of the processing required to display pictures on the Storage Display Monitor. It is the intermediary between the CPU and the Storage Display Monitor. Graphics information, when first entered in the 4081, is processed by the CPU. The CPU creates display lists from this information. The display lists can be viewed as picture programs in memory that are executed by the display processor. From the display lists, the display processor produces signals that direct the Storage Display Monitor's writing beam to draw pictures on the screen. If displayed in storage, the picture is drawn only once. If displayed in refresh, the picture must be continuously drawn and redrawn on the screen.

By handling display operations, the display processor reduces the load on the CPU, allowing faster and more efficient processing.

The Storage Display Monitor

The Storage Display Monitor allows the operator to view graphics or alphanumeric information on its 19-inch direct view storage crt (cathode ray tube). It eases communication between the operator and the 4081. The operator can view information as it is typed on the keyboard and the 4081 can display system status and error messages.

For graphics applications, the Storage Display Monitor provides a combined refresh and storage display. Stored and refresh pictures can be displayed at the same time. A stored picture is drawn once on the screen and remains visible until the screen is erased. A refresh picture is not "stored" on the screen. The picture must be continually drawn and redrawn on the screen to remain visible. In both cases, pictures are bright and clear and free from flicker.

The Keyboard Unit

The 4081 Keyboard Unit includes an alphanumeric keyboard, 12 programmable function keys, four prompt lights, an audio indicator, and a Joystick.

The alphanumeric keyboard enables an operator to talk with the 4081. By typing characters on the keyboard, the operator can initiate special 4081 functions, run programs, and enter information into memory.

The 12 function keys on the top part of the Keyboard Unit are programmable. Each key can be assigned a specific function by a program running on the 4081 or a program running on

a host computer in communication with the 4081. For example, a program can enable an operator to enlarge a picture on the screen by pressing a function key or to rotate the picture by pressing a different function key. The programs 4014 Emulator, Digitizer, and the IGT Office demonstration program all assign functions to the function keys. Each of these programs comes with an overlay that slides into the railing above the function keys. The overlay is a means to label the function assigned to each key.

The four prompt lights are red light-emitting diodes (LED's) located on the upper right of the Keyboard Unit. The four prompt lights are programmable. The lights can be programmed to function as indicators to the operator. GOS automatically assigns functions to each of the prompt lights when no other program is running on the 4081. For example, the first light, labeled REQUEST INPUT, is lit when GOS is ready to accept characters typed on the keyboard.

The audio indicator is referred to in this manual as the keyboard bell. It actually is an electronic signal played through a speaker mounted in the Keyboard Unit. The keyboard bell serves several functions under GOS. It can indicate an error condition, prompt an operator response, or signal the end of an operation. The keyboard bell, for example, rings after the GOS program is loaded into memory.

The Joystick is located on the right side of the Keyboard Unit. It looks like a stick shift for a very small sports car. It is an 8-position switch that allows the operator to locate and move pictures displayed on the screen. The Joystick is generally used to manipulate a crosshair cursor displayed on the screen. Any picture created by the user, however, can be programmed to replace the standard crosshair cursor (see the HAND program in the section **Demonstration Programs**).

The Joystick is a graphic input device. A graphic input device allows the operator to transmit X,Y coordinates to a graphics application program. The Joystick converts points on the display screen chosen by the operator into digital X,Y coordinates that can be processed by the 4081. The operator presses the JOYSTICK TERMINATOR button above the Joystick to send the X,Y coordinates of the cursor's location on the screen to the 4081. Pressing the JOYSTICK TERMINATOR button is similar to pressing the RETURN key to send a line of keyboard input to the 4081.

The Cartridge Tape Unit

The Cartridge Tape Unit, located in the left cabinet of the 4081, is a device for the long-term storage of information and programs. It enables the 4081 to read information from and write information to a tape cartridge inserted in the unit. The Cartridge Tape Unit also contains circuitry that aids in loading the data on the GOS IPL (Initial Program Load) tape cartridge into memory.

The Cartridge Tape Unit gives the 4081 the ability for local storage. Local storage means that the 4081 already has access to programs and graphics information without having to transfer them from a host computer. The operator inserts the tape cartridge that contains the desired information in the Cartridge Tape Unit. Then, by running the appropriate GOS utility programs, the operator can copy the information on the tape into the 4081 memory. It is also possible, by running the appropriate program, to copy information in the 4081 memory onto a tape cartridge. The information is then safely stored and available for future access.

The Communications Interface

The 4081 communications interface allows the 4081 to communicate with a host computer or external device. Connecting the communications interface to a host computer gives the 4081 access to the resources of a large computer system when a graphics application requires it.

Connecting the communications interface to an external device provides additional means for the input and output of graphics information processed by the 4081. The communications interface, for example, can be connected to a line printer or a plotter.

The 4081 communications interface can be connected only to host computers and devices that are RS-232-A or RS-232-C compatible. For host communications, the communications interface can be connected to any standard full-duplex modem that is RS-232 compatible. Since the communications interface is asynchronous, no external clock signal is required to synchronize the transfer of information.

THE SOFTWARE

The term software refers to programs designed to simplify the operation and programming of a computer system. The standard 4081 Graphic System software includes:

- The Graphic Operating System (GOS)
- GOS Utility Programs
- The 4014 Emulator
- The Interactive Graphics Terminal (IGT) Package
- The 4081 Verification Program
- Digitizer
- Demonstration Programs

The Graphic Operating System (GOS)

GOS is the operating system for the 4081. An operating system consists of a group of software routines that supervise the use and operation of a computer. By handling the many routine functions required to perform a task on the 4081, GOS eases communication between the user and the machine.

GOS can be viewed as a mutual interpreter between an operator and the 4081. The operator actually communicates a request directly to GOS, then GOS directs the 4081 to process the request. Instead of having to send a series of ones and zeroes to the 4081 CPU, GOS allows the operator to type requests on the keyboard in the form of simple English words and abbreviations. For example, to run a program on the 4081, the operator generally need only type the name of the program then press the RETURN key.

GOS also makes the 4081 programmer's job easier. Through device independence, GOS allows the programmer to input data from and output data to a device without regard to the device's individual quirks. The programmer can treat devices that share the same general characteristics as if they were identical. And, since GOS is designed especially for graphics applications, the programmer is supplied with the routines necessary to create, display, and manipulate pictures.

The GOS program is stored on the GOS IPL (Initial Program Load) tape cartridge.

GOS Utility Programs

The GOS utility programs facilitate many of the routine tasks of a 4081 operator. To run a utility program, the operator types the program name on the keyboard, then presses the RETURN key.

Many of the utility programs concern common file operations. The programs allow the operator to copy, rename, and delete files, to display a listing of the files stored on a tape cartridge or disc, and to display the contents of a file.

Other utility programs allow the operator to draw the contents of a file on a plotter, to print the contents of a file on a line printer, to set the data communications options for host computer communications, and to display a help message that explains the purpose and syntax of a GOS utility program.

THE STANDARD 4081 GRAPHIC SYSTEM

The GOS utility programs are stored on the system utilities tape cartridge. The programs include:

- ATTRIB
- BATCH
- COPY
- DELETE
- DIR
- DISPLA
- FORMAT
- HELP
- PDBDMP
- PLOT
- PRINT
- RENAME
- SET
- SQUISH
- SYSTAT
- TYPE

The 4014 Emulator

While the 4014 Emulator program is running on the 4081, the 4081 Graphic System emulates a TEKTRONIX 4014-1 Computer Display Terminal that contains the Enhanced Graphics Module (EGM). Emulation means the 4081 imitates a 4014 terminal so that host computer programs designed for the 4014 can be used with the 4081.

The 4014 Emulator allows 4081 operators with a library of PLOT 10 software packages and programs written with PLOT 10 software, like the PLOT 10: Terminal Control System (TCS), to use these programs with the 4081. The PLOT 10 programs can also be expanded to take advantage of the additional capabilities offered by the 4081.

The 4014 Emulator program is stored on the system utilities tape cartridge. Before running the program, the 4081 must be properly connected to a host computer.

The Interactive Graphics Terminal (IGT)

The IGT software package allows an application program running on a host computer to initiate the execution of graphics-related operations on the 4081. The 4081, in this case,

acts as an intelligent graphics terminal. Though the 4081 is under the control of the host computer, much of the processing required by the application program is performed by the 4081 itself.

The IGT package comes in two parts:

- The IGT program
- The IGT FORTRAN Host Support Subroutines

The IGT program runs on the 4081. The IGT FORTRAN Host Support Subroutines are resident on the host computer. By calling IGT FORTRAN subroutines, an application program running on the host computer can send command requests and graphics information to the IGT program for processing by the 4081.

An application program, for example, can send IGT the information necessary to create and display a picture on the 4081. The program can then modify the picture by moving, enlarging, shrinking, or rotating it. The program can also display the picture in either storage or refresh. If there is an operator at the 4081, the program can enable the operator to modify the picture using the Joystick, Graphics Tablet, or function keys.

The application program can also set the character size on the 4081's display screen, make a hard copy of the image displayed on the screen, erase the screen, save graphics information in a file on the 4081, or send graphics information back to the host computer.

The IGT program includes a formatted communications protocol that ensures the reliability of transmitted data and eases communications between different host computers and the 4081.

The 4081 Verification Program

The 4081 Verification Program is a sequence of programs that test the functioning of the 4081 Graphic System's hardware components. If a device passes the Verification Program test, the device is working properly. If a device fails the test, it could be malfunctioning. In this case, contact the local Tektronix Field Office.

The 4081 Verification Program should be run at least once a month for the purpose of routine maintenance. The tests include:

- The Initial Program Load (IPL) Function Test
- The Prompt Lights Test
- The Memory Test
- The Processor Test
- The Storage Display Monitor Test
- The Keyboard Unit Test
- The Cartridge Tape Unit Test
- The Communications Interface Test

For an expanded 4081 Graphic System, these additional tests are available:

- The Flexible Disc Unit Test
- The Hard Disc Unit Test
- The Graphics Tablet Test
- The Plotter Test
- The Line Printer Test

The 4081 Verification Program comes on two tape cartridges.

Digitizer

Digitizer can be run only on a 4081 Graphic System that includes one of the optional TEKTRONIX 4953/4954 Graphics Tablets. The Digitizer program allows an operator to draw a picture on the Graphics Tablet while the picture-in-progress is displayed on the 4081 Storage Display monitor. The graphics information for the picture can then be stored in a file on the 4081 for future display and editing.

The name Digitizer refers to the process of digitizing, converting the points of a picture into digital X,Y coordinates that can be processed by the 4081. Digitizing is a feature of graphic input devices like the Graphics Tablet.

The Digitizer program is stored on the system utilities tape cartridge.

Demonstration Programs

The demonstration programs that come with the standard 4081 Graphic System provide an entertaining way to get acquainted with the operation and capabilities of the 4081. The programs illustrate several important 4081 features - the clarity of displayed pictures, combined storage and refresh graphics, picture manipulation, and resident processing power.

The demonstration programs are stored on the system utilities tape cartridge. The programs include:

- LOGO
- XLOGO
- LANDER
- CUBE
- HAND
- MOTOR
- LIFE



SECTION 4

EXPANDING THE 4081 GRAPHIC SYSTEM

CONTENTS

	Page
The Hardware	4-3
The 4905 Mass Storage Module	4-7
The 4631 Hard Copy Unit	4-7
The 4641 Character Printer	4-8
The 4662 Interactive Digital Plotter	4-10
The 4953/4954 Graphics Tablets	4-12
The Hexadecimal Display Panel	4-14
The PROM Bootstrap Loader	4-16
An Additional 32K Bytes of Memory	4-17
An Additional Cartridge Tape Unit Drive	4-17
The Expansion Package	4-17
The Software	4-18
The Programming Support Package	4-19
The Assembler	4-19
GOS TECO (Text Editor and Corrector)	4-20
The Library Linker/Loader	4-20
RAID Debugging Aids	4-20
The FORTRAN IV Compiler	4-21
The Compiler	4-21
The Run Time Library	4-21

Section 4

EXPANDING THE 4081 GRAPHIC SYSTEM

The standard 4081 Graphic System can be expanded to handle a wider variety of graphics applications. The optional hardware and software available allow the user to customize the 4081 to fit the application.

THE HARDWARE

All of the optional hardware is designed to connect easily to the 4081. In addition, controlling routines for most of the devices are already available in the Graphic Operating System (GOS) program.

The optional 4081 Graphic System hardware (see Fig. 4-1) includes:

- The 4905 Mass Storage Module (with Flexible Disc and/or Hard Disc Units)
- The 4631 Hard Copy Unit
- The 4641 Character Printer
- The 4662 Interactive Digital Plotter
- The 4953/4954 Graphics Tablets
- The Hexadecimal Display Panel
- The PROM Bootstrap Loader
- An Additional 32K Bytes of Memory
- An Additional Cartridge Tape Unit Drive
- The Expansion Package

EXPANDING THE 4081 GRAPHIC SYSTEM

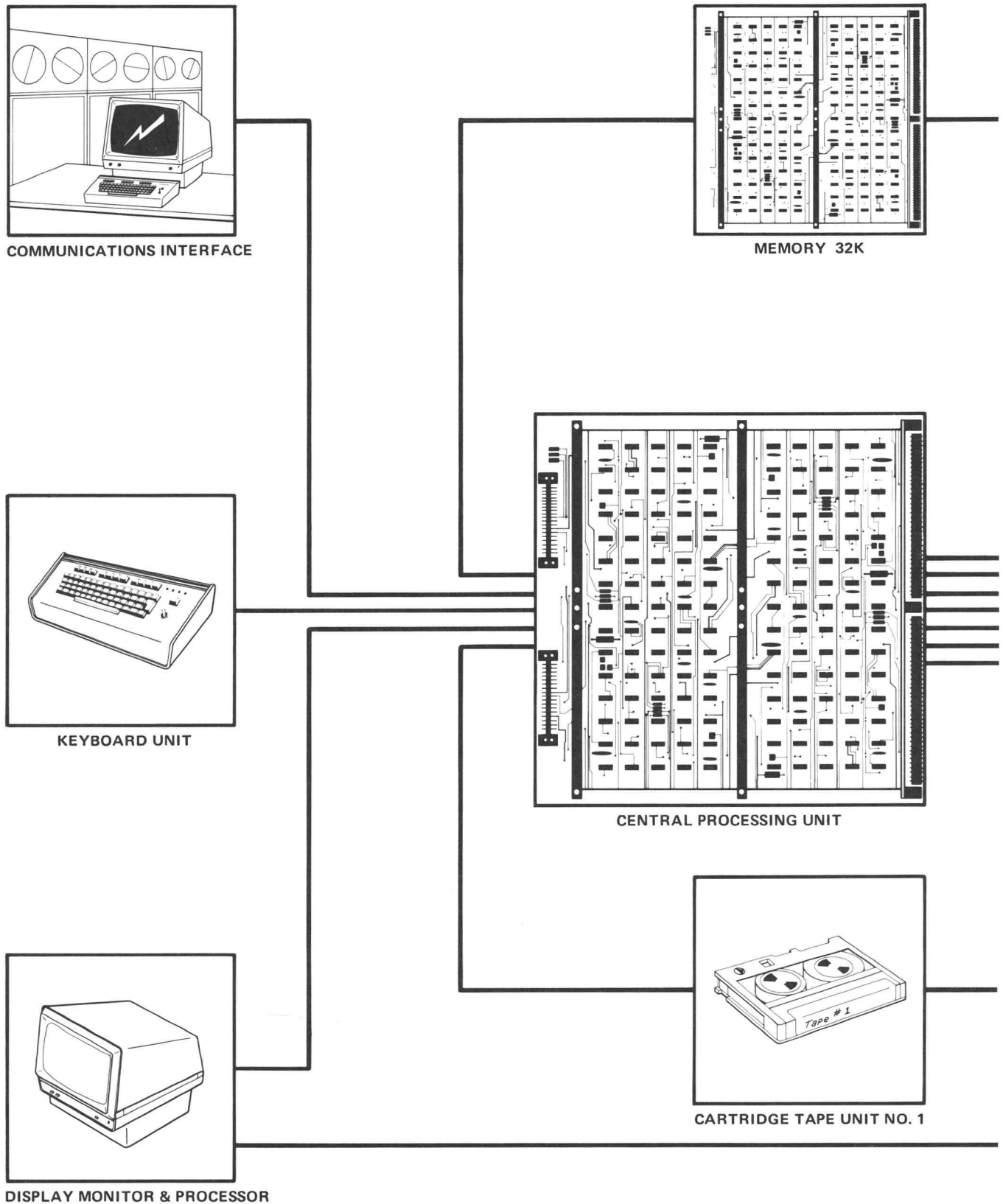


Fig. 4-1. The expanded 4081 Graphic System.

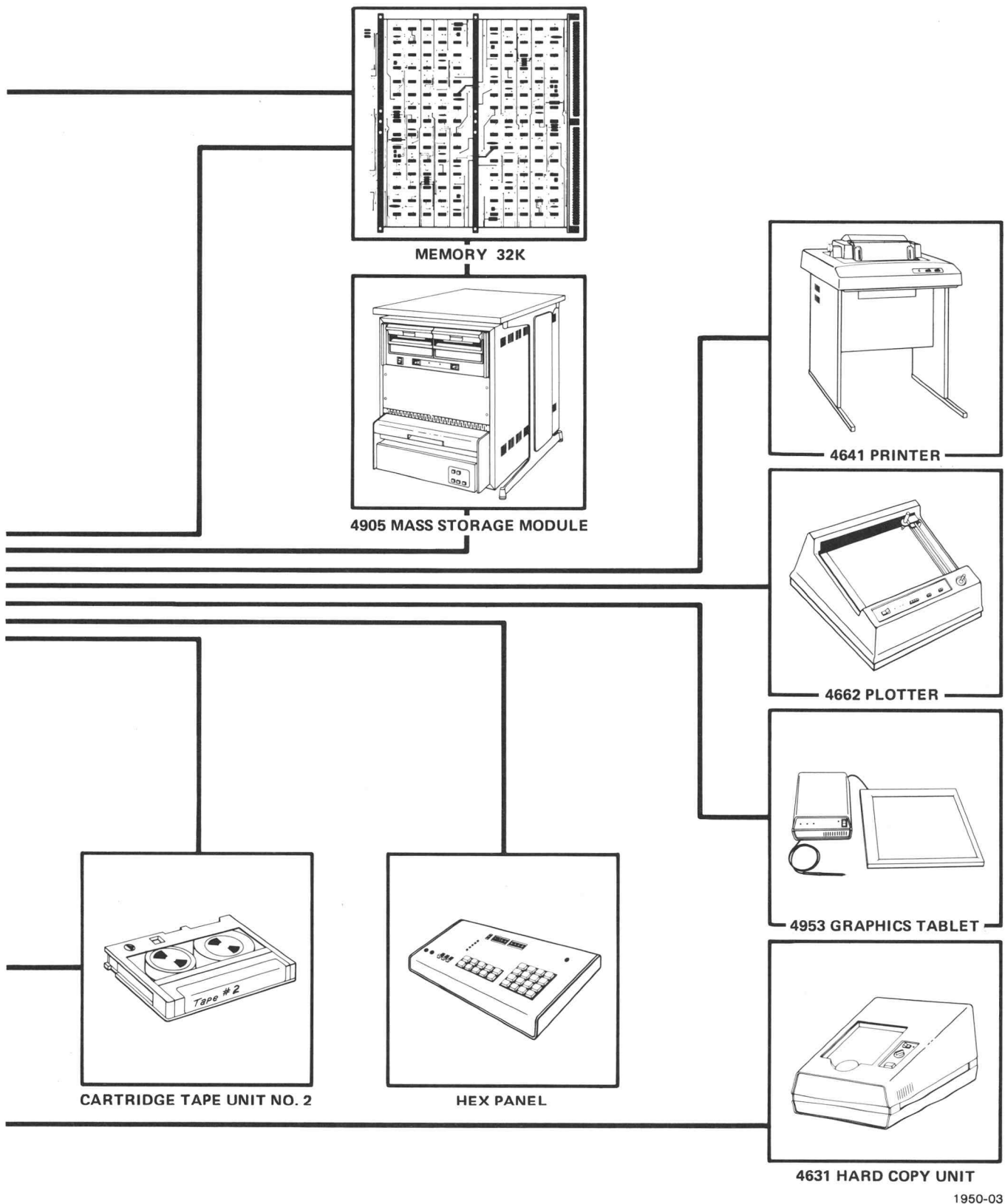
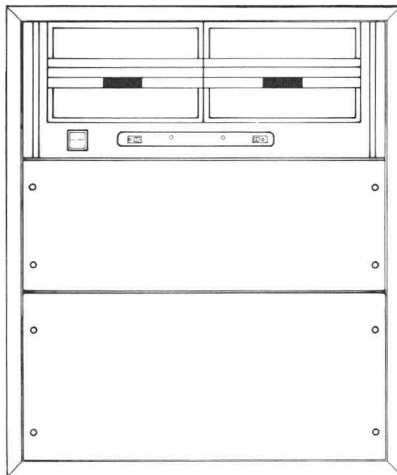
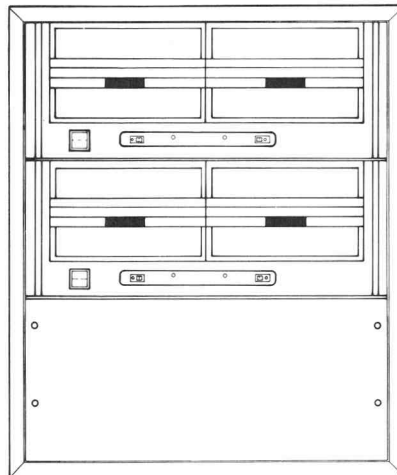


Fig. 4-1. The expanded 4081 Graphic System (cont).

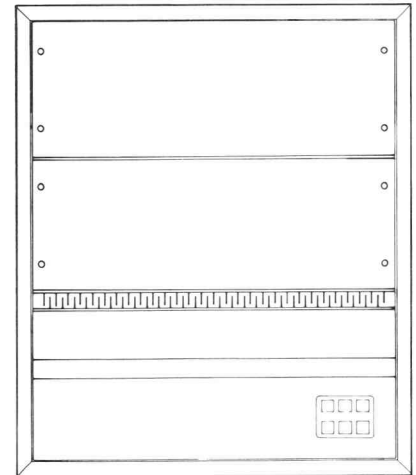
EXPANDING THE 4081 GRAPHIC SYSTEM



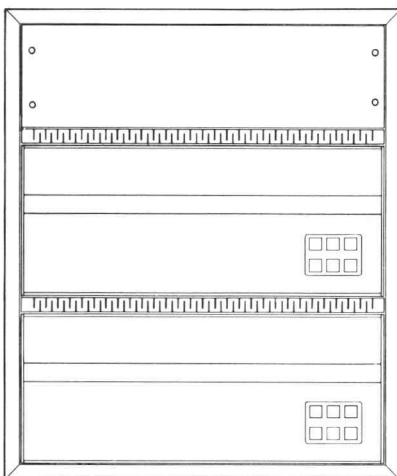
OPTION 31
One Flexible Disc Unit



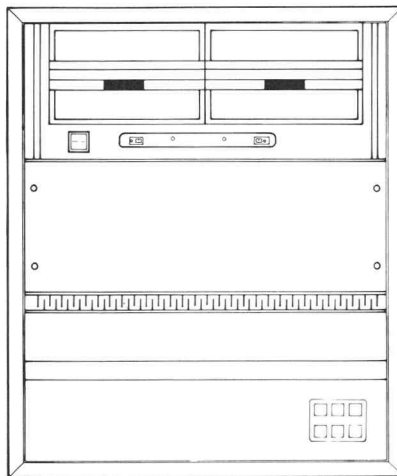
OPTION 32
Two Flexible Disc Units



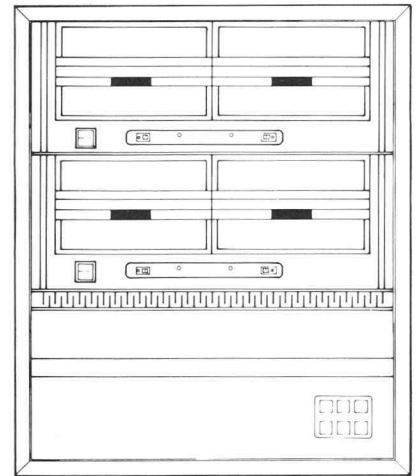
OPTION 33
One Hard Disc Unit



OPTION 34
Two Hard Disc Units



OPTION COMBINATION 31, 33
One Flexible Disc Unit and
One Hard Disc Unit



OPTION COMBINATION 32, 33
Two Flexible Disc Units and
One Hard Disc Unit

(2269)1950-04

Fig. 4-2. The six options available for the 4905 Mass Storage Module.

The 4905 Mass Storage Module

The 4905 Mass Storage Module is a cabinet that contains up to two Flexible Disc Units, up to two Hard Disc Units, or combinations of the two devices (Fig. 4-2 shows the six available options).

Both the Flexible Disc Unit and the Hard Disc Unit are, like the Cartridge Tape Unit, devices for the long-term storage of information and programs. They enable the 4081 to read information from and write information to a flexible disc or hard disc inserted in the appropriate unit.

The Flexible Disc Unit allows faster file processing and has a greater capacity for data storage than the Cartridge Tape Unit. The Hard Disc Unit allows faster file processing and has a greater capacity for data storage than either the Flexible Disc or Cartridge Tape Units. (For further information on the three devices, see **File Structured Devices** in the section **Files**.)

All GOS utility programs that perform file-related functions (for example, COPY, DELETE and FORMAT) can be used with the 4905 Mass Storage Module.

A 4905 with at least one Flexible Disc or Hard Disc Unit is a prerequisite for the Plot 80: Programming Support Package and the Plot 80: FORTRAN IV Compiler.

The 4631 Hard Copy Unit

The 4631 Hard Copy unit (Fig. 4-3) makes a permanent, high-contrast copy of an image displayed on the 4081 Storage Display Monitor. To produce a copy, the operator presses the HARD COPY button on the front panel of the Display Monitor or the COPY button on the Hard Copy Unit. Under GOS, a hard copy also can be produced by a program running on the 4081.

When activated, the Hard Copy Unit makes a complete sweep of the Display Monitor, copying all pictures currently displayed in storage. (Pictures displayed in refresh cannot be copied by the Hard Copy Unit.) After the screen image is applied to the hard copy paper, the paper is automatically cut and the image is heat-developed within the unit. The paper copy is then ejected into the paper tray in the top of the unit. The entire process takes only several seconds.



Fig. 4-3. The 4631 Hard Copy Unit.

The 4641 Character Printer

The 4641 Character Printer (Fig. 4-4) is an alphanumeric line printer. It provides a permanent, typewriter-like copy of alphanumeric information stored on the 4081. Any sequence of text that can be displayed on the 4081 Storage Display Monitor can be printed on the Printer.

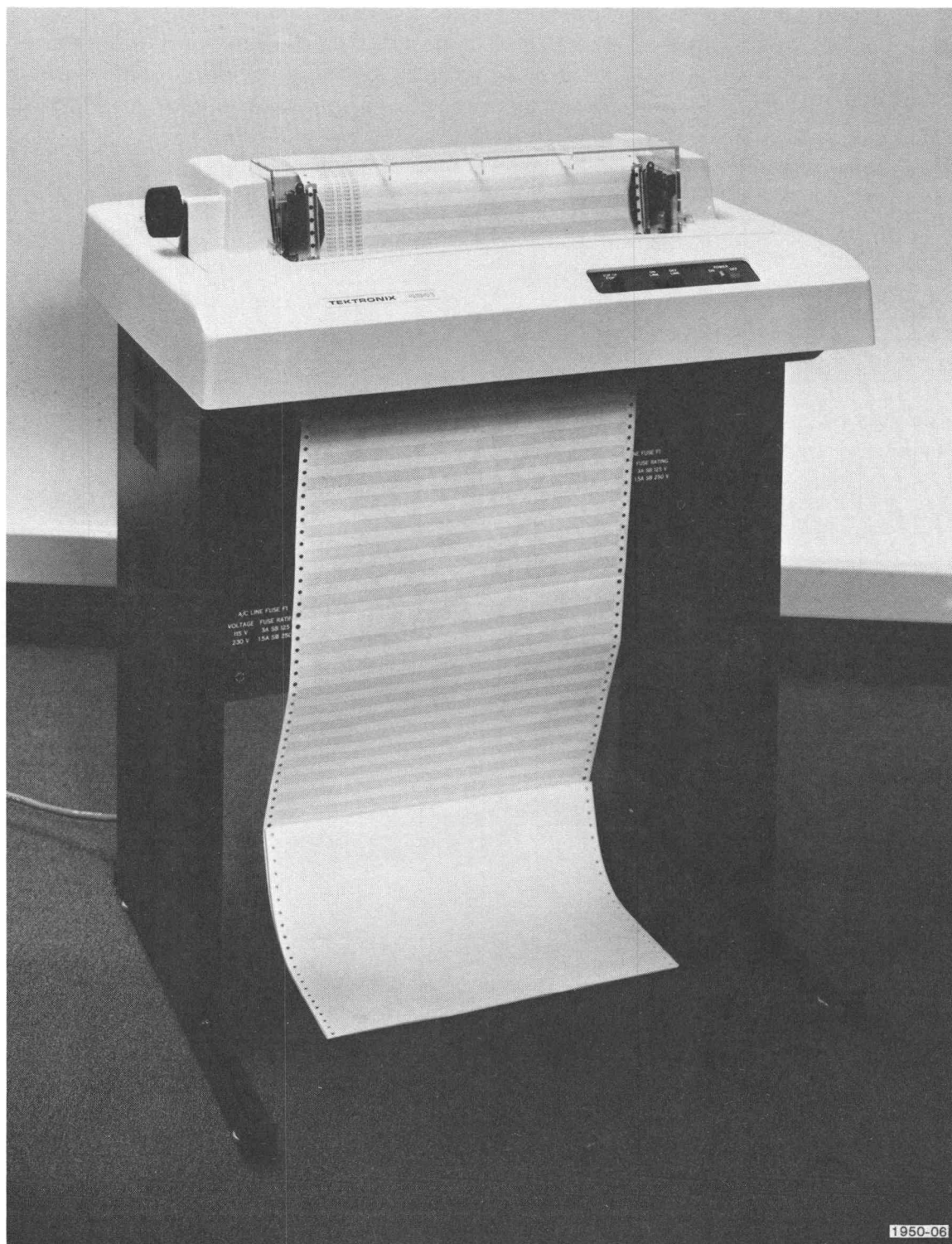


Fig. 4-4. The 4641 Character Printer.

The GOS utility program PRINT allows an operator to print the contents of a 4081 ASCII file on the Printer. ASCII files contain alphanumeric information like text or program source code stored in the form of ASCII binary code. While an ASCII file is printed on the Printer, a program can be running on the 4081 in the background (in other words, when the Printer is not using the 4081's processor). Therefore, printing a long sequence of text on the Printer does not tie up the 4081 for long periods. The PRINT utility program can be run only on a 4081 with 64K bytes of memory (the additional 32K bytes of memory is available as an option).

The 4662 Interactive Digital Plotter

The 4662 Interactive Digital Plotter (Fig. 4-5) provides a permanent copy of both alphanumeric and graphics information stored on the 4081. Any picture or sequence of text that can be displayed on the 4081 Storage Display Monitor can be drawn on the Plotter.

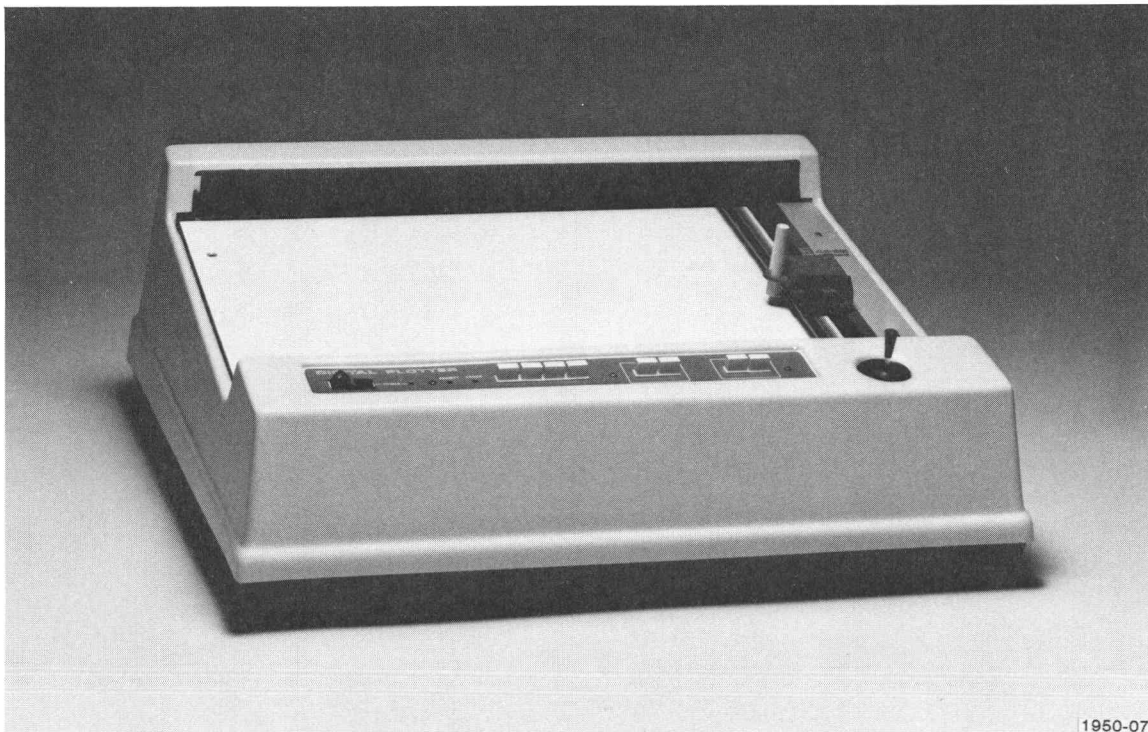


Fig. 4-5. The 4662 Interactive Digital Plotter.

The GOS utility program PLOT allows an operator to draw the contents of a 4081 graphics file on the Plotter. Graphics files contain picture information stored in the form of screen X,Y coordinates. While a graphics file is drawn on the Plotter, a program can be running on

the 4081 in the background (in other words, when the Plotter is not using the 4081's processor). Therefore, drawing a complex or large picture on the Plotter does not tie up the 4081 for long periods. The PLOT utility program can be run only on a 4081 with 64K bytes of memory (the additional 32K bytes of memory is available as an option).

The Plotter is also, like the Joystick and Graphics Tablet, a graphic input device. A graphic input device allows the operator to transmit X,Y coordinates to a graphics application program. The Plotter converts points on its plotting surface chosen by the operator into digital X,Y coordinates that can be processed by the 4081.

The Plotter pen is used to locate points of a picture attached to the plotting surface. The pen also can be used to locate points of a picture displayed on the 4081 Display Monitor. In this case, moving the pen also moves a cursor displayed on the screen.

Generally, when the PROMPT light on the Plotter is lit, a program running on the 4081 is requesting graphic input from the operator. The operator uses the Plotter's Joystick to move the pen over the surface of the Plotter. When the pen is in the desired position, the operator presses the CALL button to send the X,Y coordinates of the pen's position to the 4081. The PROMPT light is turned off when the 4081 receives the coordinates. If the Plotter bell rings, the CALL button may have been pressed too long and no coordinates were sent to the 4081. In this case, press the CALL button again. Pressing the CALL button is similar to pressing the JOYSWITCH TERMINATOR button when using the Joystick for graphic input.

Before operating the Plotter, make sure that the four dials marked A, B, C and D on the rear panel are set correctly. The correct setting for each dial is shown in Fig. 4-6.

The dial settings indicate selectable options for operating the Plotter. The options selected include:

Data Transfer Rate:	1 200 baud
Plotting Speed:	High
Parity:	None

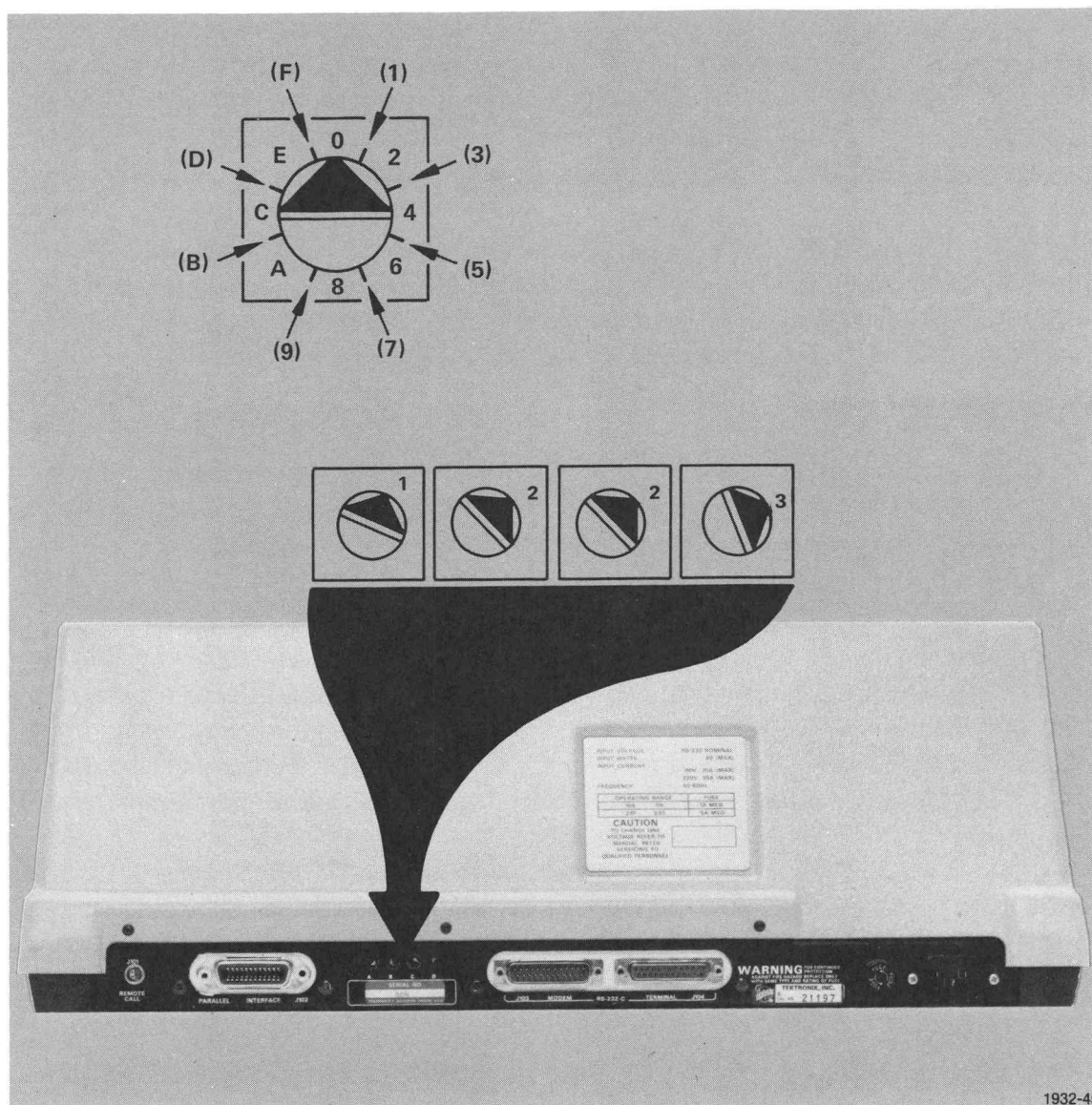
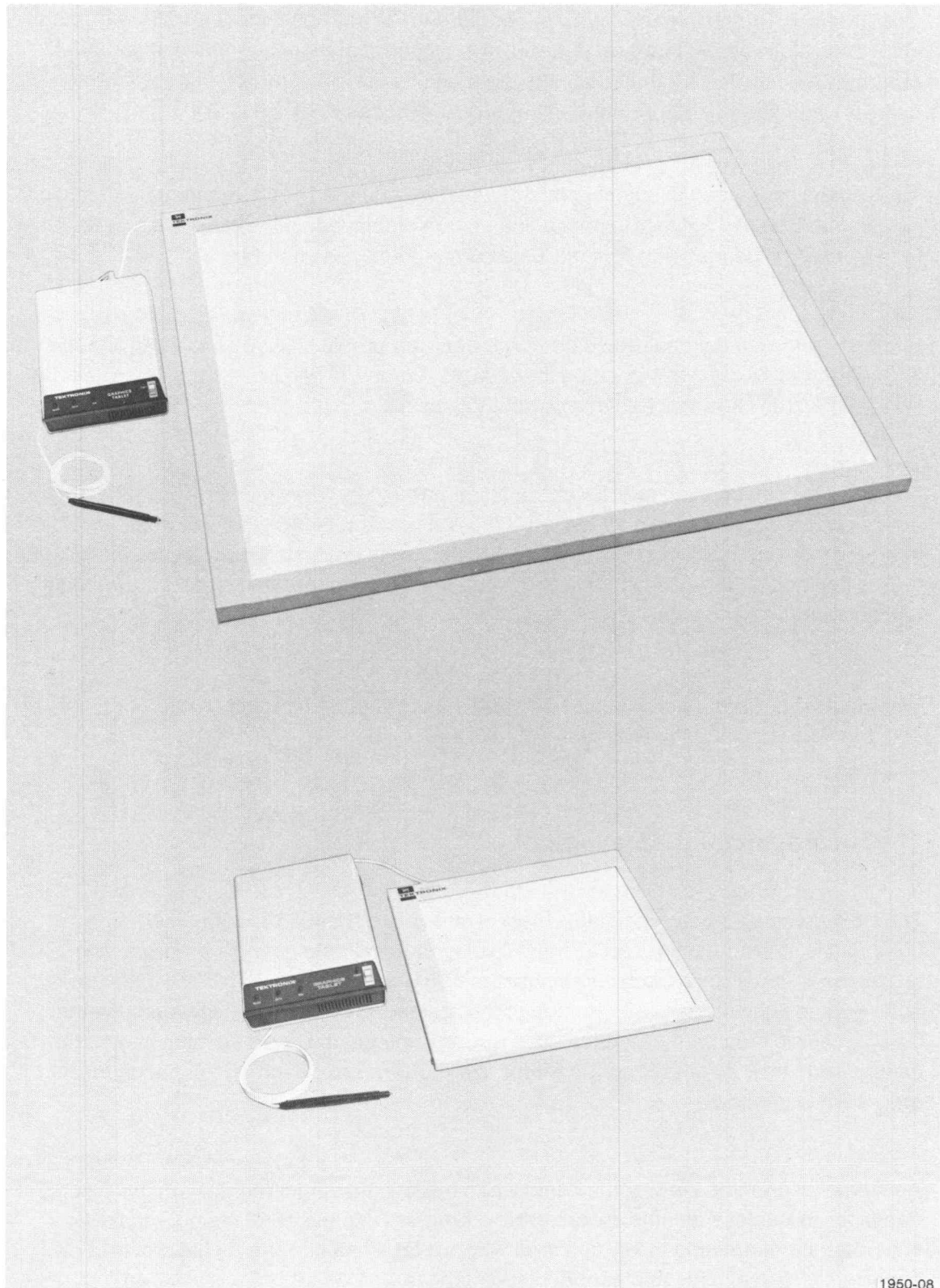


Fig. 4-6. The Plotter dial settings for operation with the 4081.

The 4953/4954 Graphics Tablet

The 4953/4954 Graphics Tablets (Fig. 4-7) provide a means to enter graphics information into the 4081. The 4953 Graphics Tablet is the small tablet. It has a drawing surface of 11 inches by 11 inches. The 4954 Graphics Tablet is the large tablet. Its drawing surface measures 40 inches by 30 inches. The tablets can be ordered separately.



1950-08

Fig. 4-7. The 4953 and 4954 Graphics Tablets.

The Graphics Tablet, like the Joystick and Plotter, is a graphic input device. A graphic input device allows the operator to transmit X,Y coordinates to a graphics application program. The Graphics Tablet converts points on its drawing surface chosen by the operator into digital X,Y coordinates that can be processed by the 4081.

The Tablet pen is used to locate points of a picture attached to the drawing surface. The pen also can be used to locate points of a picture displayed on the 4081 Display Monitor. In this case, moving the pen also moves a cursor displayed on the screen.

When the pen is in the desired position, the operator presses down on the pen to send the X,Y coordinate of the pen's position to the 4081. When the pen is pressed down, the red PEN light on the Graphics Tablet's power module is lit.

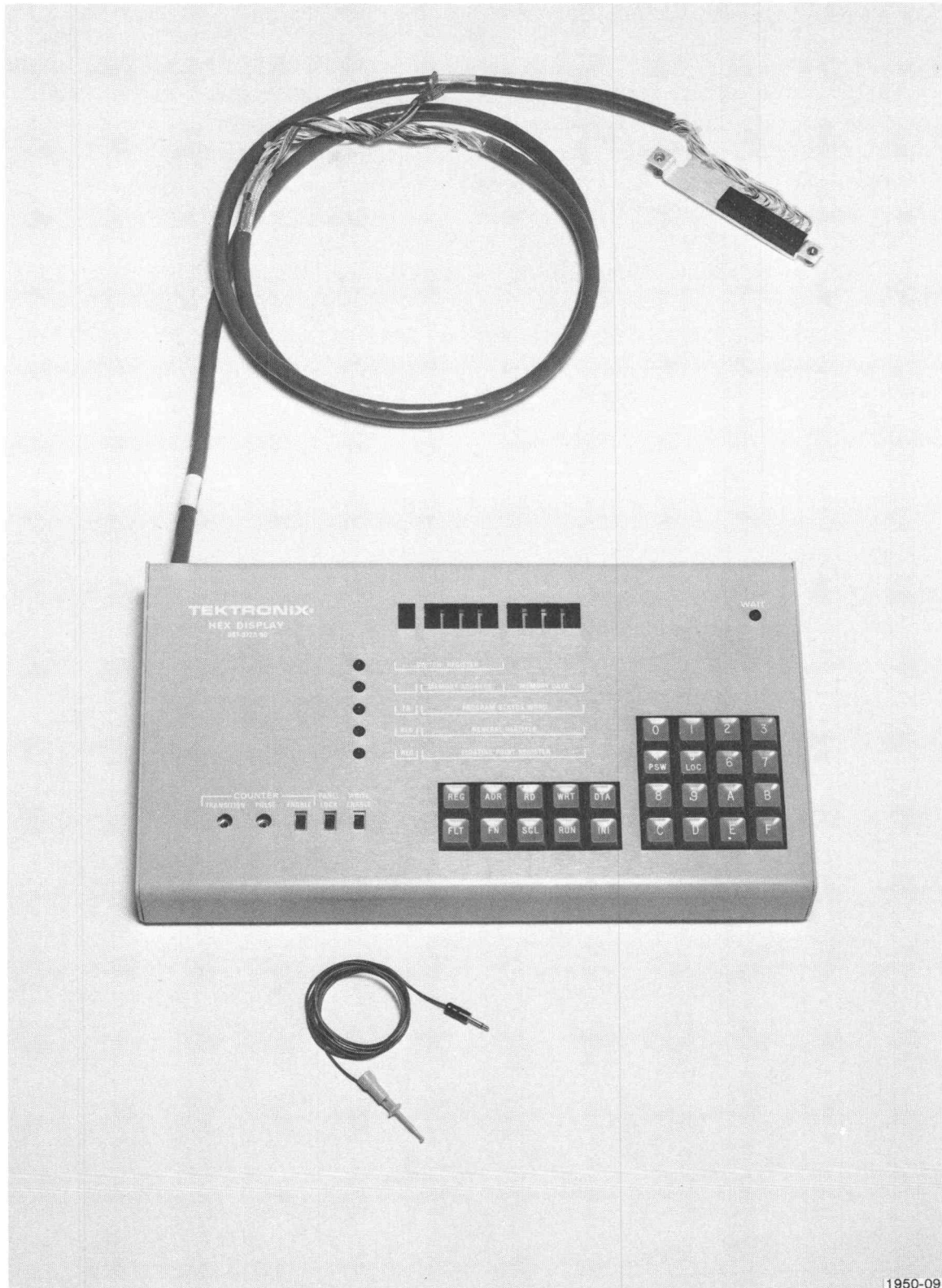
The pen also can be used to draw or trace a picture on the drawing surface of the Tablet. As the picture is being drawn or traced, it is displayed on the 4081 Display Monitor. The operator must continually press down on the pen while drawing. The graphics information for the finished picture, once in the 4081 memory, can be stored in a file for future display and editing.

The Digitizer program, which comes with the standard 4081 Graphic System, provides a good demonstration of the Graphics Tablet's capabilities.

The Hexadecimal Display Panel

The Hexadecimal Display Panel (Fig. 4-8) is an aid to both 4081 programmers and technicians. A programmer can use the Display Panel to help debug a program. The programmer first enters the program into the 4081 memory by using the Display Panel or other means, then starts the execution of the program. The Display Panel enables the programmer to monitor the program as it runs, examining the contents of memory, the 16 general registers, and the Program Status Word (PSW) after each instruction in the program is executed.

A technician can use the Display Panel when troubleshooting circuits. With the Display Panel, for example, a technician can enter a program into the 4081 memory that loops continuously, exercising the circuit area where a fault has occurred. This provides an opportunity to use test equipment to pinpoint the fault.



1950-09

Fig. 4-8. The Hexadecimal Display Panel.

The PROM Bootstrap Loader

The PROM Bootstrap Loader, or Romulan Loader, (Fig. 4-9) is a special purpose circuit card that plugs into the card cabinet of the 4081. The card loads programs or data into the 4081 memory. The card has four sockets for mounting Programmable Read-Only Memory (PROM) devices on which the program or data to be loaded is permanently stored.

Two PROM devices come with the PROM Bootstrap Loader. The IPL (Initial Program Load) PROM contains a program that locates the Graphic Operating System (GOS) program or any other operating system program for the 4081 and loads it into the 4081 memory. The operating system program must be stored on a file on a hard disc or flexible disc inserted in the 4905 Mass Storage Module.

The other PROM device contains a 4081 Graphic System diagnostic program that tests the Memory, the Keyboard Unit, the Central Processing Unit (CPU), and the Hexadecimal Display Panel.

The PROM Bootstrap Loader also can be used with user-programmed PROMs.

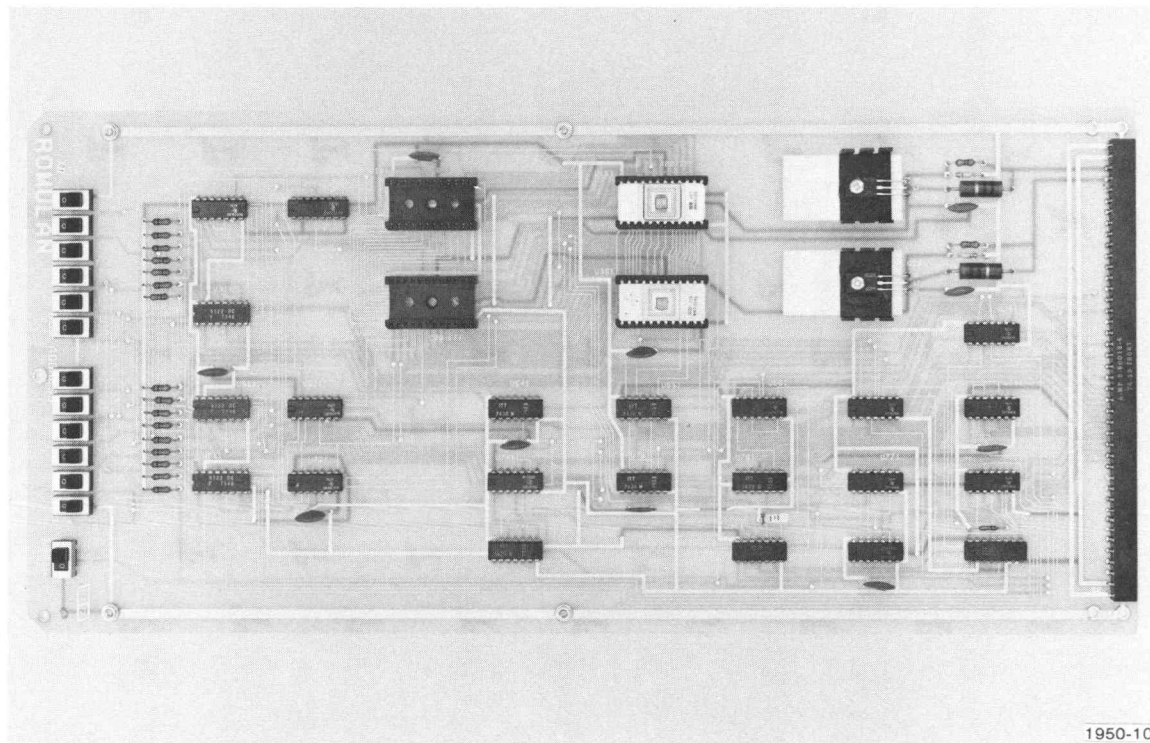


Fig. 4-9. The PROM Bootstrap Loader.

An Additional 32K Bytes of Memory

An additional 32K bytes of memory are available to supplement the 32K bytes that come with the standard 4081 Graphic System. Additional memory allows the 4081 to execute larger programs and display more complex refresh pictures.

The additional memory is a prerequisite for the Plot 80: Programming Support Package and the Plot 80: FORTRAN IV Compiler.

An Additional Cartridge Tape Unit Drive

An additional Cartridge Tape Unit drive increases the local data storage capacity of the 4081. The additional drive gives the 4081 access to two tape cartridges at the same time. For example, the system utilities tape cartridge can be inserted in the standard first Cartridge Tape Unit drive while a tape cartridge containing the user's program and data files is inserted in the optional second Cartridge Tape Unit drive.

The additional Cartridge Tape Unit drive is installed in the left cabinet of the 4081, below the standard Cartridge Tape Unit drive.

All GOS utility programs that perform file-related functions (for example, COPY, DELETE and FORMAT) can be used with the additional Cartridge Tape Unit drive.

The Expansion Package

The Expansion Package includes an additional power supply for the 4081 and a secondary backplane with eight full circuit card slots.

The standard 4081 Graphic System contains eight full circuit card slots. The circuit cards for the standard 4081 occupy 5 1/2 slots. There are 2 1/2 slots remaining for expanding the 4081. If the optional devices ordered for a 4081 occupy more than 2 1/2 circuit card slots, then the Expansion Package is required.

Table 4-1 shows the number of circuit card slots required for each optional device available:

Table 4-1
REQUIRED CIRCUIT CARD SLOTS

Device	No. of Slots
4905 Mass Storage Module	
With One Flexible Disc Unit (Option 31).....	1
With Two Flexible Disc Units (Option 32).....	2
With One Hard Disc Unit (Option 33).....	2
With Two Hard Disc Units (Option 34)	2
With One Flexible Disc Unit and	
One Hard Disc Unit (Options 31 and 33).....	3
With Two Flexible Disc Units and	
One Hard Disc Unit (Options 32 and 33).....	4
4631 Hard Copy Unit	None
4641 Character Printer	1/2
4662 Interactive Digital Plotter	1/2
4953/4954 Graphics Tablets.....	1/2 per Tablet
Hexadecimal Display Panel.....	None
PROM Bootstrap Loader	1/2
An Additional 32K Bytes of Memory.....	1
An Additional Cartridge Tape Unit Drive	None

THE SOFTWARE

All of the optional software is designed to run on the 4081 under the control of the Graphic Operating System (GOS) program.

The optional 4081 Graphic System software includes:

- The Programming Support Package
- The FORTRAN IV Compiler

THE PROGRAMMING SUPPORT PACKAGE

The Programming Support Package contains the software and documentation necessary to write, edit, assemble, debug, and run programs on the 4081. The software that comes with the Programming Support Package includes:

- The Assembler
- GOS TECO (Text Editor and Corrector)
- The Library Linker/Loader
- RAID Debugging Aids

In addition to documentation for the software listed above, the Programming Support Package includes documentation detailing the features and routines that GOS offers the programmer.

The Assembler

A program to be run on the 4081 is written using the Plot 80: Assembly Language. Assembly language provides symbols and mnemonics to represent machine code instructions. Machine code consists of combinations of binary digits. Each combination, when received by a computer's processing unit, initiates a specific function.

The Assembler translates a program source file that contains a sequence of Plot 80: Assembly Language statements into a program load file that can be loaded into the 4081 memory and executed. In other words, the Assembler translates the symbols and mnemonics used to write a program into a binary code that can be directly understood by the 4081.

The Assembler features include macro instructions, list-processing instructions, pseudo-ops (special instructions to the Assembler that are not executed by the 4081), the symbolic representation of 4081 memory locations and the 16 general registers, and software-emulated floating-point capabilities.

GOS TECO (Text Editor and Corrector)

GOS TECO provides a means to create and edit program source files. TECO enables a programmer to create a file, insert program statements and data into the file, then edit the file if any changes to the program are necessary.

GOS TECO also can be used to create and edit files that contain alphanumeric text other than program source code. For example, using TECO, an operator can type a document on the 4081 keyboard and store the information in a file. The operator can then display the contents of the file on the 4081 Storage Display Monitor while editing the text for typing errors or revisions. When the document is completed, the operator can make a permanent copy by printing the file on a line printer.

The Library Linker/Loader

The Library Linker/Loader provides a means to load, link and execute programs. Remember that the Assembler translates a program source file that contains a sequence of Plot 80: Assembly Language Statements into a program load file. The program load file then must be loaded into the 4081 memory and executed. In addition, some programs consist of several modules that may reside in separate program load files. The separate program load files must be loaded into memory and linked into one program before the program can be executed.

The Library Linker/Loader performs the functions of loading into memory the separate modules that make up a program, linking the modules, then initiating the execution of the program.

The Linker also features an overlay facility. If a program is too large to load into the 4081 memory at once, the Linker divides it into sections (or overlays). The overlays are then loaded into memory during program execution as they are needed.

RAID Debugging Aids

RAID Debugging Aids enables a programmer to locate and correct errors in a program. Using RAID, a programmer can stop the execution of a program running on the 4081 at predefined points in the program. The programmer can then examine the contents of memory and the 16 general registers, displaying the contents on the Display Monitor in hexadecimal, decimal, alphanumeric, floating point, and assembly language formats.

If an error is detected in a program, RAID allows the programmer to deposit different statements or data into the program, then run the incorrect section of the program again to verify the change.

NOTE

The Programming Support Package requires a 4081 Graphic System with 64K bytes of memory and a 4905 Mass Storage Module with at least one Flexible Disc or Hard Disc Unit.

THE FORTRAN IV COMPILER

The FORTRAN IV Compiler enables a programmer to compile and run FORTRAN programs on the 4081. The Plot 80: FORTRAN IV Compiler package combines ANSI Standard FORTRAN statements with GOS-related routines.

The FORTRAN IV Compiler package includes:

- The Compiler
- The Run Time Library

The Compiler

The Compiler translates a program source file that contains a sequence of Plot 80: FORTRAN IV statements into a program load file that can be loaded into the 4081 memory and executed.

The Run Time Library

The Run Time Library contains a set of GOS-related FORTRAN routines. The routines can be called by a FORTRAN program to make full use of a 4081 Graphic System running under the control of GOS.

Generally, the routines are either related to file and device functions or graphics functions. The graphics-related routines are designed to allow a programmer to create, display, and manipulate graphics information on the 4081.

The Run Time Library also includes several routines that allow a running program to initiate common 4081 operations, like erasing the screen or making a hard copy of the information displayed on the Display Monitor.

The Run Time Library routines called by a program are loaded into the 4081 memory and linked with the program load file before the program is run. The Library Linker/Loader that comes with the Plot 80: Programming Support Package performs the loading and linking functions.

NOTE

The FORTRAN IV Compiler requires a 4081 Graphic System with 64K bytes of memory, a 4905 Mass Storage Module with at least one Flexible Disc or Hard Disc Unit, and the Plot 80: Programming Support Package.

SECTION 5

A SESSION WITH THE 4081

CONTENTS

	Page
Basic Equipment for the Session.	5-3
Basic Assumptions for the Session	5-4
The Session.....	5-4

Section 5

A SESSION WITH THE 4081

This section is designed to guide new 4081 operators through their first session on the 4081. Several of the GOS-assigned keyboard characters, resident commands, GOS utility programs, and related concepts discussed in detail in other sections of this manual are presented as they might be used in real life.

The practice session includes many of the 4081 functions most useful to an operator: turning on the 4081, loading the GOS program into the 4081 memory, running programs, setting the display character size, and editing characters typed on the keyboard.

The session does not contain detailed explanations of the concepts and functions presented. It provides only a basic idea of what an operator can do on the 4081 and how to do it.

Detailed explanations, however, are available in the other sections of this manual. Refer to the table of contents or the index to locate the desired information.

Copies of the 4081 display screen shown in the session were produced by the 4631 Hard Copy Unit.

BASIC EQUIPMENT FOR THE SESSION

- 4081 Graphic System
- 4905 Mass Storage Module, Option 31
(2 Flexible Disc Unit drives)

For a 4081 with a Hard Disc Unit drive, replace all references in the session to the Flexible Disc Unit drive(FD0:) with the Hard Disc Unit drive(DK0:).

For a 4081 with only a Cartridge Tape Unit drive, replace all references in the session to the Flexible Disc Unit drive(FD0:) with the top Cartridge Tape Unit drive(CT0:).

BASIC ASSUMPTIONS FOR THE SESSION

- The 4081 Graphic System is installed and running properly.
- The 4905 Mass Storage Module is properly connected to the 4081.
- The system disc containing the GOS utility programs is properly inserted in the first Flexible Disc Unit drive.

THE SESSION

Power

Turn on the 4081 by pressing the POWER switch located on the front panel of the left cabinet. The green POWER light on the front panel of the Display Unit is lit. The main circuit breaker on the back of the 4081 must be in the ON position.

And More Power

If the 4905 Mass Storage Module is off, turn it on by pressing the POWER switch located on the front panel of the Flexible Disc Unit. The green indicator light in the switch is lit. The MAIN POWER switch on the lower rear side of the 4905 should be in the ON position and the green light in the switch lit.

The Brains Behind the Machine

The Graphic Operating System program (GOS) is a set of software routines that allow an operator to easily communicate with the 4081. The operator actually communicates directly with GOS, then GOS directs the 4081 to process the operator's requests. GOS can be viewed as a mutual interpreter between an operator and the 4081.

The GOS program is stored on the GOS IPL (Initial Program Load) tape cartridge, a standard part of the 4081 Graphic System. The GOS program must be loaded into the 4081 memory before the 4081 can begin executing it.

Some Cheap Insurance

Before using the GOS IPL tape cartridge, make sure that it is write-protected. Write-protecting the cartridge prevents the operator from accidentally writing over the GOS program stored on the tape.

To write-protect the GOS IPL tape cartridge, first find the black arrowhead on the top side of the cartridge. The black arrowhead should be pointing to the position marked SAFE. If the arrowhead points away from the SAFE position, insert a screwdriver or coin into the slot behind the arrowhead. Turn the arrowhead until it locks into the SAFE position. The GOS IPL tape cartridge is now write-protected.

Loading GOS

When the 4081 power is turned off, the 4081 memory is cleared. When the 4081 is turned on again, the operator must load the GOS program into memory before attempting to communicate with the 4081.

To load the GOS program into memory, first insert the GOS IPL tape cartridge into the Cartridge Tape Unit drive in the upper half of the left cabinet. The cartridge automatically rewinds to the beginning of the tape. Notice that the red BUSY light is lit.

When the tape stops moving (the BUSY light is off), press the IPL button. The GOS program is now copied into memory from the tape cartridge. When the complete GOS program is loaded, the display screen is erased, the keyboard bell rings, and the following is displayed:

```
GOS V3 0L0
#
```

GOS V3 0 L0 are the version and level numbers of the GOS program.

The **#** is a GOS prompt character that indicates the 4081 is ready to receive instructions to execute a resident command or program file. The operator types the instructions on the keyboard. Notice that the red REQUEST INPUT prompt light on the upper right of the keyboard is lit. The light is lit whenever the 4081 is ready to receive characters typed on the keyboard.

A Free Option

On some 4081's the operator may notice additional information displayed with the GOS version and level numbers and the # prompt character. This additional information results from a convenient feature available as a GOS program option. If this option is selected, then as soon as the GOS program is loaded into the 4081 memory, GOS automatically executes a sequence of commands that have been loaded into memory as if typed on the keyboard. In most cases, the only command loaded into memory as part of the GOS IPL procedure is the utility program BATCH. The BATCH program causes a sequence of commands stored in a batch file to be executed on the 4081.

The batch file is a file on the system utilities disc. The commands stored in the batch file are selected by the user. The commands can be any combination of GOS resident commands, GOS utility programs, and user programs. The Tektronix field representative, when creating the user's GOS IPL tape cartridge, includes a command to GOS to run the BATCH program and execute the user's batch file. This option is available as part of the standard 4081 Graphic System.

The user's batch file generally contains the commands most often typed by an operator after the 4081 is turned on and the GOS program is loaded into memory. By including the commands in the batch file, GOS executes the commands automatically, saving the operator the trouble of typing them. For example, if a 4081 is used mainly to communicate with a host computer, the user might include a command in the batch file to initialize the primary communications interface (SET COM).

In this session, the sample batch file contains two commands. The first command asks the operator to type the date on the keyboard, then press the RETURN key. The second command asks the operator to type the time of day on the keyboard, then press the RETURN key. The sample batch file is called SYS:INIT.BAT. The user, however, may select any name for the batch file.

Loading GOS Again

Now, for 4081's that include the automatic BATCH option, again load the GOS program into the 4081 memory by pressing the IPL button. When the complete GOS program is loaded, the display screen is erased, the keyboard bell rings, and, this time, the following is displayed:

```
GOS V3 0L0
#DD-MMM-YY
*
```


The batch file SYS:INIT.BAT, when executed by GOS, first asks the operator to type the date in the form **DD-MMM-YY**. **DD** is the date, **MMM** is the first three letters of the month, and **YY** is the last two digits of the year. For example, if the date is December 25, 1977, type,

* 25-dec-77

then press the RETURN key.

```
GOS V3 0L0
#DD-MMM-YY
*25-DEC-77
HH-MM-SS
*
```

Now the batch file SYS:INIT.BAT asks the operator to type the time of day in the form **HH-MM-SS**. **HH** is the hour, **MM** is the minutes, and **SS** is the seconds. For example, if the time is 1:23 and 45 seconds in the afternoon, type,

* 1-23-45 PM

then press the RETURN key.

```
GOS V3 0L0
#DD-MMM-YY
*25-DEC-77
HH-MM-SS
*1-23-45 PM
->End of Batch Stream
#
```

The ring of the bell and the message **->End of Batch Stream** means that all the commands in the batch file have been executed. The current data and time are now entered in the 4081 memory. As long as the 4081 is on, the time is constantly updated by a clock-like circuit.

A Mysterious Experiment

To illustrate an important GOS concept, remove the system disc from the first Flexible Disc Unit drive (the left-most drive on the 4905 Mass Storage Module) and insert it in the second Flexible Disc Unit drive (the right-most drive). Make sure that the drive door is shut

and the drive is write-protected. Write-protecting the drive prevents the operator from accidentally writing over the utility program files stored on the disc. To write-protect the drive, flip the black WRITE PROTECT switch on the drive toward the red WRITE PROTECT light. The light should be lit.

Now, again load the GOS program by pressing the IPL button. When the complete GOS program is loaded, the display screen is still erased, the keyboard bell still rings, but this time, the following is displayed:

```
GOS V3 0L0
#Dir I/O error DU lu 16
#
```

Dir I/O error DU lu 16 is an error message that means, in this case, GOS attempted to run a program file but the medium containing the program file was missing. (**DU** means device unavailable.) The missing program file is the BATCH utility program.

Was It Something I Said?

GOS was able to run the BATCH program when the GOS program was loaded into memory the first time. Why did GOS display an error message this time?

The error results from removing the system disc from the first Flexible Disc Unit drive. GOS looked for the BATCH program file on the disc inserted in this drive. Unfortunately, there was no disc inserted in this drive. Why didn't GOS look for the BATCH program file on the disc inserted in the second Flexible Disc Unit drive?

Setting Things Right

To save some time in this experiment, the operator can run the BATCH program instead of again loading the GOS program into memory and having GOS run the BATCH program automatically. To run the BATCH program exactly as GOS runs it, type,

```
#batch sys:init.bat
```

then press the RETURN key.

```
#batch sys init bat
Dir I/O error DU lu 16
#
```

Notice that GOS displays the same error message as before.

Now type,

```
#fd1:set sys:=fd1:
```

then press the RETURN key.

A clicking sound comes from within the second Flexible Disc Unit drive and the red BUSY UNIT light is lit. When the prompt character # is displayed on the screen, again type,

```
#batch sys:init.bat
```

then press the RETURN key.

```
#fd1 set sys =fd1
#batch sys init bat
DD-MMM-YY
*
```

No error message is displayed this time. The BATCH program is running as it did before, first asking the operator to type the date.

The date and time typed earlier in the session are no longer in the 4081 memory. Remember that since last typing the date and time, the GOS program was again loaded into memory. Loading the GOS program into memory destroys all the data that was stored in memory before the GOS program was loaded.

So, type the date and time again as shown earlier in the session.

A File By Another Name

The operator may be wondering what went right. Why did the BATCH program fail twice to run, then run the third time? Notice that after the system disc was inserted in the second Flexible Disc Unit drive, the BATCH program ran only after typing,

```
# fd1:set sys:=fd1:
```

fd1: is a mnemonic for the second Flexible Disc Unit drive. Each device connected to the 4081 has its own mnemonic. The mnemonic is just a quick and easy way of referring to a device. It takes less time to type "FD1:" on the keyboard than "Flexible Disc Unit drive 1".

fd1:set is a program file on the system disc. The full name of the file is FD1:SET.OBJ. There are three distinct parts of the file's name: the device on which the file is located (FD1:), the filename (SET), and a filename extension that further identifies the file (.OBJ).

When the operator types the name of a file after the prompt character **#** is displayed on the screen, GOS assumes the file is a program file. GOS attempts to load the program file into the 4081 memory and run it.

By typing,

```
#fd1:set sys:=fd1:
```

GOS attempts to load the program file FD1:SET.OBJ into memory and run it. Notice that if no filename extension is typed for a program file, GOS assumes a default filename extension of .OBJ.

Now, in the example,

```
#batch sys:init.bat
```

GOS attempts to run a program file called BATCH. However, the device part of the file's name was not typed. On what device is the BATCH program file located? GOS needs this information to find the file.

Alias SYS:

Remember that no filename extension was typed when running the program file FD1:SET.OBJ. GOS assumed a default filename extension of .OBJ. GOS also assumes a default when the device part of a program file's name is omitted. The default device for a program files is SYS:.

What is SYS:? Is it a Cartridge Tape Unit, a Flexible Disc Unit, or a Hard Disc Unit? It could be any of these devices. SYS: is a nickname, an alias, a temporary label for a file-structured device. The mnemonic SYS: is assigned to a device. Generally, it is assigned to the device containing the system utility programs. When running a program file that is stored on the device assigned the SYS: mnemonic, the operator does not have to type the device part of the file's name. The SYS: mnemonic is a convenience to the operator.

So, back to BATCH. The program file BATCH.OBJ is on the system disc inserted in the second Flexible Disc Unit drive (better known as FD1:). By typing,

```
#fd1:set sys:=fd1:
```

the operator assigned the SYS: mnemonic to the second Flexible Disc Unit drive. SYS: becomes just another way of referring to FD1:.

Now, when the operator types,

```
# batch sys:init.bat
```

GOS loads the program file SYS:BATCH.OBJ into memory and runs it. SYS: is the default device and .OBJ is the default filename extension. The three following ways of running the BATCH program file are identical to GOS:

```
# batch
```

```
# sys:batch.obj
```

```
# fd1:batch.obj
```

The Case is Closed

After the operator removed the system disc from the first Flexible Disc Unit drive (FD0:) and inserted it in the second Flexible Disc Unit drive (FD1:), the BATCH program failed to run when the GOS program was loaded into memory.

The reason for the failure is that, on most 4081 systems that include a Flexible Disc Unit, the GOS program automatically assigns the SYS: mnemonic to FD0:. (The device to which the SYS: mnemonic is assigned is determined by the user when the GOS IPL tape cartridge is created.)

When GOS attempted to run the program file SYS:BATCH.OBJ, it looked for the file on the disc inserted in FD0:. The system disc, however, was inserted in FD1:. GOS couldn't find the BATCH program file and displayed an error message.

By typing,

```
# fd1:set sys:=fd1:
```

the operator reassigned the SYS: mnemonic to FD1:. Now when GOS attempted to run the program file SYS:BATCH.OBJ, it looked for the file on the disc inserted in FD1:. Because the system disc was inserted in FD1:, GOS was able to find the file and run the program.

A Return to Normalcy

The GOS program, when loaded into the 4081 memory, generally assigns the SYS: mnemonic to the first Flexible Disc Unit drive. So, to simulate actual operating conditions for the rest of the session, remove the system disc from FD1: and insert it back in FD0:. Make sure the drive door is closed and the red WRITE PROTECT light is lit.

Now reassign the SYS: mnemonic to FD0: by typing,

```
#fd0:set sys:=fd0:
```

then press the RETURN key.

Running Programs

The programs BATCH and SET are GOS utility programs. The GOS utility programs are stored in program files on the system disc. Now that the system disc is inserted in the first Flexible Disc Unit drive, it's possible to try running a few more of the GOS utility programs.

In the following examples, when a GOS utility program is run, notice that the device and filename extension parts of the utility program file's name are omitted. Since all the utility program files have an .OBJ extension and GOS assumes a default extension of .OBJ for program files, the filename extension can be omitted. Also, since FD0: is assigned the mnemonic SYS: and GOS assumes a default device of SYS: for program files, the device can be omitted.

Do You Have the Time?

In the beginning of this session, the operator typed the date and time of day on the 4081. How can the operator make sure that the 4081 received this information and has it stored in memory?

Type,

```
# systat
```

then press the RETURN key.

#systat

System Status of GOS V3 0L0

December 25, 1977 13 27 5

Devices KB,DC,JOY,CT0,CT1,FD0,FD1,FD2,FD3,TB0(Sm),CM0*,PL0
 NUL,SYS(FD0),USR(FD1),GIN(JOY)

Comm 300 Baud, Matched rates, No Parity
 Local Echo, 2 Stop bit(s), 200ms Break

Character size 5, 64KB memory, COSTOP - 6FD0
 Refresh objects - 16

#

Running the SYSTAT utility program displays the 4081 system status information on the screen. The system status information includes the date and time typed earlier by the operator. Notice that the time of day has been automatically updated by the 4081.

The system status information also includes other information of interest to an operator. Notice the device mnemonics listed after the heading **Devices**. The mnemonic SYS: is listed on the second line. Listed in parentheses after SYS: is the device to which SYS: has been assigned. Remember that SYS: was assigned to the first Flexible Disc Unit drive(FD0:).

Also listed is another reassignable device mnemonic, USR:. The USR: mnemonic usually is assigned to the device containing the 4081 user's own files. Like the SYS: mnemonic, the USR: mnemonic is a convenience to the operator.

A Useful Device

To run the SYSTAT utility program, the operator typed,

#systat

The default device for the SYSTAT program file is SYS:.

Under GOS, the operator can also run the SYSTAT utility program by typing,

#run sys:systat

A SESSION WITH THE 4081

Notice that now the operator has to type SYS:SYSTAT, not just SYSTAT. The default device for the program file is no longer SYS:. When running a program file by typing the RUN command, the default device for the program file is USR:.

The operator, however, can reassign the USR: mnemonic to the device that contains the SYSTAT program file. Since the SYSTAT program file is in the first Flexible Disc Unit drive(FD0:), type,

```
#set usr:=fd0:
```

then press the RETURN key.

Now the operator can run the SYSTAT program file by typing the RUN command and still omit the device. Type,

```
#run systat
```

then press the RETURN key.

```
#set usr =fd0  
#run systat
```

System Status of GOS V3 0L0

December 25, 1977 13 28 18

Devices KB,DC,JOY,CT0,CT1,FD0,FD1,FD2,FD3,TB0(Sm),CM0*,PL0
 NUL,SYS(FD0),USR(FD0),GIN(JOY)

Comm 300 Baud, Matched rates, No Parity
 Local Echo, 2 Stop bit(s), 200ms Break

Character size 5, 64KB memory, COSTOP - 6FD0
Refresh objects - 16

■

Notice that now both the SYS: and USR: mnemonics are assigned to the first Flexible Disc Unit drive(FD0:).

Remember, typing,

```
#systat
```

is the same as typing,

```
#sys:systat.obj
```

and typing,

```
#run systat
```

is the same as typing,

```
#run usr:systat.obj
```

Information, Please

The system disc contains a directory at the beginning of the disc. The directory includes a list of all the files stored on the disc. By running the DIR utility program, the operator can display the directory on the screen. Type,

```
#dir sys:
```

then press the RETURN key.

```
mdir sys
```

```
Directory Structure of SYS SYSTEM 25-Dec-77
Directory Blocks Available- 8, Used- 3
```

DIR	OBJ	[17]	24-May-77	14	43	
SET	OBJ	[34]	24-May-77	14	54	
SYSTAT	OBJ	[8]	24-May-77	14	49	
4014	OBJ	[14]	24-May-77	14	56	
DIGIT	OBJ	[20]	24-May-77	14	52	
IGT	OBJ	[61]	10-Mar-77	2	58	
DEMO	LIB	[180]	30-Jul-76	11	45	L
DELETE	OBJ	[8]	24-May-77	14	42	
ATTRIB	OBJ	[8]	24-May-77	14	41	
RENAME	OBJ	[11]	24-May-77	14	42	
COPY	OBJ	[13]	24-May-77	14	45	
TYPE	OBJ	[8]	24-May-77	14	46	
DISPLA	OBJ	[8]	24-May-77	14	47	
PRINT	OBJ	[2]	24-May-77	14	51	
PLOT	OBJ	[2]	24-May-77	14	51	
SPOOL	SAV	[4]	17-May-77	9	51	
FORMAT	OBJ	[22]	24-May-77	14	47	
SQUISH	OBJ	[10]	24-May-77	14	50	
BATCH	OBJ	[3]	24-May-77	14	51	
HELP	OBJ	[4]	24-May-77	14	50	
HELP	LIB	[120]	15-May-77	13	19	L
INIT	BAT	[1]	23-May-77	15	18	

```
Files-22 ,Blocks-558 ,Free-666 ,Largest-666
```

```
#
```

Most of the files listed in the directory are GOS utility program files. Notice the programs run previously in this session: BATCH, SET, SYSTAT, and DIR. The second column of the directory lists the filename extension for each file. Generally, the program files have an .OBJ extension.

Help

The directory listing shows an imposing collection of GOS utility programs. GOS, however, does not abandon 4081 operators to figure out how to use the programs on their own. Even without referring to the section on GOS utility programs in this manual, an operator can view a short message describing the syntax and function of each utility program. Type,

```
#help
```

then press the RETURN key.

```
#help
Help Version 3 0 Level 0

HELP NAME[,LISTDEV]
Default extension is HLP
Help is available for
```

```
Directory Structure of SYS HELP    15-May-77
Directory Blocks Available= 2, Used= 2
```

DISPLA	HLP	[3]	13-May-77	15	39
SQUISH	HLP	[6]	9-Mar-77	14	23
TYPE	HLP	[3]	9-Mar-77	14	25
RENAME	HLP	[6]	13-May-77	15	57
DIR	HLP	[12]	13-May-77	16	00
DELETE	HLP	[3]	13-May-77	16	08
ATTRIB	HLP	[6]	13-May-77	16	10
COPY	HLP	[11]	13-May-77	16	14
SYSTAT	HLP	[4]	13-May-77	16	25
BATCH	HLP	[4]	13-May-77	16	27
FORMAT	HLP	[11]	13-May-77	16	30
SET	HLP	[20]	9-Mar-77	15	35
PDBDMP	HLP	[2]	9-Mar-77	16	07
HELP	HLP	[4]	15-Apr-76		
ERRORS	HLP	[13]	16-May-77	12	54

```
Files-15,Blocks-108,Free-10,Largest-9
```

```
$
```

A list of available help messages is displayed. Notice that the last character displayed on the screen is \$. The \$ is a prompt character for the HELP program. It means that the HELP program is waiting for the operator to type a help message request.

What if the operator doesn't want to view a help message at this time? How does the operator terminate the HELP utility program?

Gone...

To interrupt a program that is running on the 4081 (in this case, the HELP utility program), press the SHIFT key and, while depressing the SHIFT key, press the ESC key.

```
$<A↑↑n>  
#
```

The HELP program now stops running. The # prompt character indicates that the 4081 is ready to receive further instructions to execute commands.

...But Not Forgotten

The HELP program, however, is not gone for good. To continue the HELP program from the point at which it was interrupted, type,

```
# con
```

then press the RETURN key.

```
$<A↑↑n>  
#con
```

Nothing has been displayed on the screen. Is the HELP program really still running?

First press the RESET/PAGE key to erase the screen. The screen is getting a little too cluttered. Now for the test. For a help message on the SYSTAT utility program, type,

```
SYSTAT
```

Then press the RETURN key.

SYSTAT**Name** SYSTAT**Format** SYSTAT [LISTDEV]

Purpose To provide the IGS user with information regarding current GOS parameters. These parameters include the date, devices included in GOS, sys, usr and gin devices, the GOS version and level number, primary communication option status (primary communications device is denoted by a "\$" character in the devices list), character size, the amount of memory, the tablet size(s), the current top of the operating system, and the number of refresh objects available to the GOS programmer.

Switches None**Wildcards** None**Default extension** "LST" if a filename is specified for a list device

Example #SYSTAT LP This will give a listing of the above mentioned GOS parameters on the line printer. If a list device(file) is not specified, the display (DC) is assumed.

•

The SYSTAT help message is displayed, followed by another \$ prompt character. Again, press the SHIFT-ESC keys to terminate the HELP program. Press the RESET/PAGE key to erase the screen.

Commands-In-Residence

The RUN and CON (CONTINUE) commands typed earlier in the session are different than GOS utility programs. The GOS utility programs are stored in files on the system disc. If the system disc is not inserted, the utility programs cannot be run.

Commands like RUN and CON, however, are resident commands. These commands are actually loaded into the 4081 memory as part of the GOS program. The operator can execute a resident command whether or not the system utilities tape cartridge, flexible disc, or hard disc is inserted.

To illustrate the concept of a resident command, remove the system disc from the first Flexible Disc Unit drive. Type,

* this is a comment

then press the RETURN key.

```
#*this is a comment
#
```

* is a resident command. Its only function is to indicate a comment line on the screen. GOS ignores a comment line. If the * command is omitted, GOS attempts to process the comment. This time omitting the * command, type,

```
# this is a comment
```

then press the RETURN key.

```
# this is a comment
SYS THIS OBJ not found!
#
```

GOS displays an error message because GOS assumed the first word on the line, "this", was a program file called SYS:THIS.OBJ. (Notice how GOS adds the default device and filename extension to "this".) GOS then attempted to run SYS:THIS.OBJ, but, of course, there is no such file.

Changing Character Sizes

The resident command CHA (CHARSIZE) allows the operator to change the size of the characters displayed on the screen. There are ten character sizes available: 1 is the smallest size and 10 is the largest. Type,

```
# cha 6
```

press the RETURN key and type,

```
# * this is a big comment
```

then again press the RETURN key.

```
# cha 6
# * this is a big comment
#
```


Now type,

```
#cha 3
```

press the RETURN key and type,

```
# *this is a small comment
```

then again press the RETURN key.

```
#cha 3
#this is a small comment
#
```

To return to the normal display screen character size, type,

```
#cha 4
```

then press the RETURN key.

A Few Key Points

Remember that when the operator pressed the RESET/PAGE key, the display screen was erased. Like the RESET/PAGE key, there are several other keys on the 4081 keyboard that have special functions.

Two keys, RUBOUT and ESC, allow the operator to edit the characters typed on the keyboard. Try typing something incorrectly. Type,

```
# * a typing errol
```

There is one mistake in the line. The word "error" is misspelled. Press the RUBOUT key. The "l" is covered with a small cross-hatched rectangle. Type an "r", the correct letter, and press the RETURN key.

```
#*a typing erro
#
```

Now try typing most of the line incorrectly. Type,

* a piping era

It would be easier to type the line again instead of using the RUBOUT key to cross out each character. Press the ESC key. is displayed directly after the incorrect line. Type the entire line again, this time correctly, below the incorrect line, then press the RETURN key.

```
#*a piping era<Del>
*a typing error
#
```

Upper and Lower Case

Throughout the session, all the characters typed on the keyboard have been displayed as lower case characters. It is possible, however, to type all upper case letters. The TTY LOCK key is similar to a shift-lock device on a typewriter. Press the TTY LOCK key, causing it to lock in the down position. Type,

* ALL CAPITAL LETTERS

then press the RETURN key.

```
#*ALL CAPITAL LETTERS
#
```

To unlock the TTY LOCK key, press it again, causing it to return to the up position. Type,

* all lower case letters

then press the RETURN key.

```
#*all lower case letters
#
```

SECTION 6

DEMONSTRATION PROGRAMS

CONTENTS

	Page
LOGO.....	6-4
XLOGO	6-6
LANDER	6-8
CUBE.....	6-10
HAND.....	6-12
MOTOR.....	6-14
LIFE	6-17

Section 6

DEMONSTRATION PROGRAMS

The seven demonstration programs that are included with the standard 4081 Graphic System illustrate several important features of the 4081. The programs are stored on the system tape (or disc) in the library file DEMO.LIB. DEMO.LIB contains the following programs:

- LOGO
- XLOGO
- LANDER
- CUBE
- HAND
- MOTOR
- LIFE

Before attempting to run any of these programs, enter Command mode by pressing the ESC key while depressing the SHIFT key. The system tape or disc must be inserted in the device that is assigned the mnemonic SYS:. (For information on assigning the SYS: mnemonic, see **SET** and **SYSTAT** in the section **GOS Utility Programs**.) Type,

DEMO/program name

then press the RETURN key.

program name is the name of the desired demonstration program.

LOGO

The LOGO program demonstrates device independence. It displays the Tektronix logo (Fig. 6-1) on the Display Monitor in storage or draws it on another designated device (such as the Plotter). It can also store the logo, without displaying it on the screen, in a designated file on a medium inserted in a mass storage device connected to the 4081. The designated file can either already exist or be created by the LOGO program. Copying the logo to a file that already exists, however, destroys the previous contents of the file.

To display the logo on the Display Monitor:

Type: **DEMO/LOGO**

Press: RETURN

To draw it on another device:

Type: **DEMO/LOGO dev:** (**dev:** is the device mnemonic)

Press: RETURN

Example: **DEMO/LOGO PL:** draws the logo on the Plotter.

To store the logo in the file TEKLOG on the user device:

Type: **DEMO/LOGO TEKLOG**

Press: RETURN

To store it in the file TEKLOG on another device:

Type: **DEMO/LOGO dev:TEKLOG** (**dev:** is the device mnemonic)

Press: RETURN

Example: **DEMO/LOGO FD1:TEKLOG** stores the logo in the file **TEKLOG** on the disc inserted in the second Flexible Disc Unit drive (**FD1:**)

The filename extension .PLT is assigned by default when the logo is stored in a designated file. The GOS utility program DISPLA displays the logo stored in the designated file.

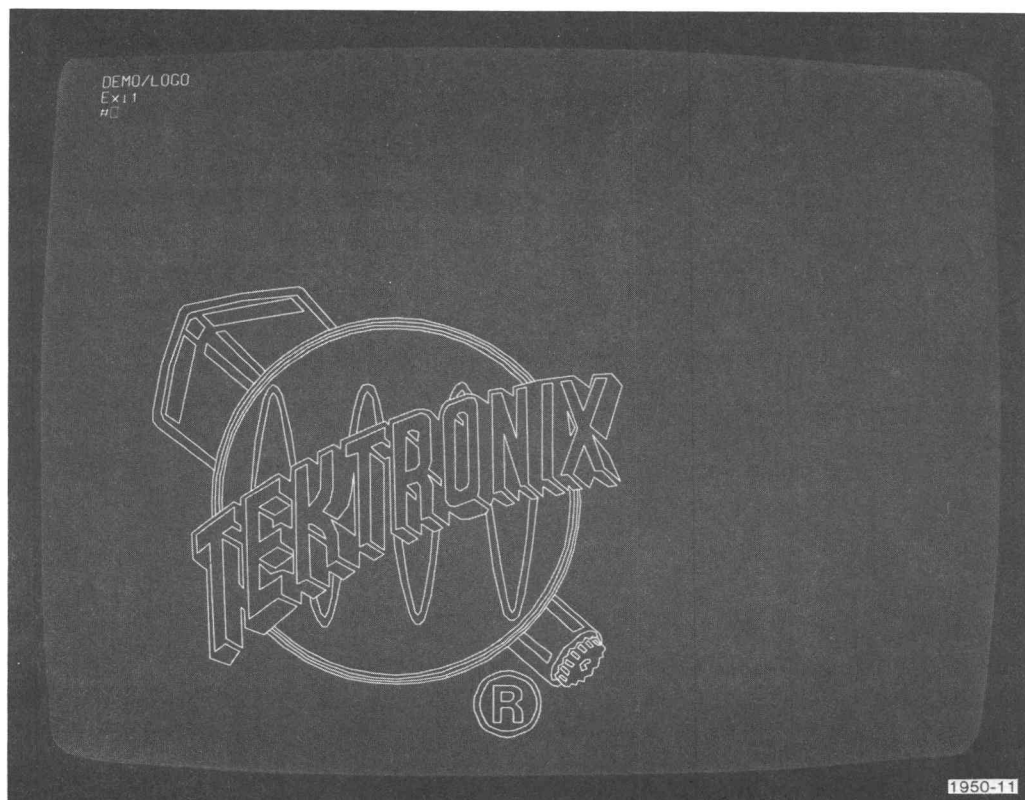


Fig. 6-1. Tektronix logo.

To display the logo stored in the file TEKLOG on the user device:

Type: **DISPLA TEKLOG**

Press: RETURN

To display the logo stored in the file TEKLOG on another device:

Type: **DISPLA dev:TEKLOG** (**dev:** is the device mnemonic)

Press: RETURN

Example: **DISPLA FD1:TEKLOG** displays the logo stored in the file **TEKLOG** on the disc inserted in the second Flexible Disc Unit drive (**FD1:**)

When the LOGO program is finished, the following appears on the screen:

Exit
#

XLOGO

The XLOGO program demonstrates the ability to refresh approximately 800 vectors, flicker free, by displaying the Tektronix logo on the Display Monitor in refresh. To display the logo:

Type: **DEMO/XLOGO**
Press: RETURN

The Joystick moves the logo about the screen. To increase the speed with which the logo travels across the screen, depress the SHIFT key while operating the Joystick. Press the JOYSWITCH TERMINATOR key to display the logo in storage at its current screen location. The logo is then displayed again in refresh and can be moved to another location and stored (Fig. 6-2a and Fig. 6-2b). The operator can repeat this process until the program is terminated.

To terminate the program and store the logo at its current location, press any alphanumeric key, then press the JOYSWITCH TERMINATOR key. The following appears on the screen when the program is terminated (x represents the alphanumeric key pressed):

xExit
#

Press the RESET/PAGE key to erase the screen. Type **STA** and press RETURN to restart the program.

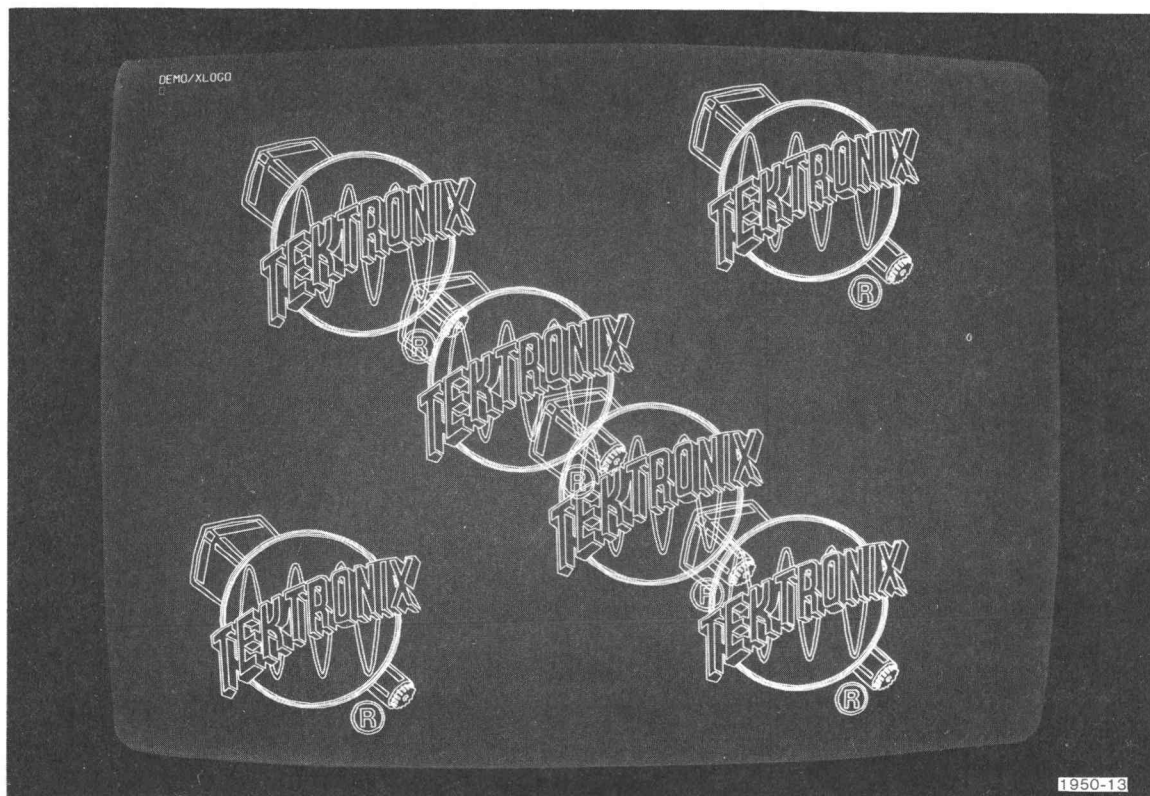
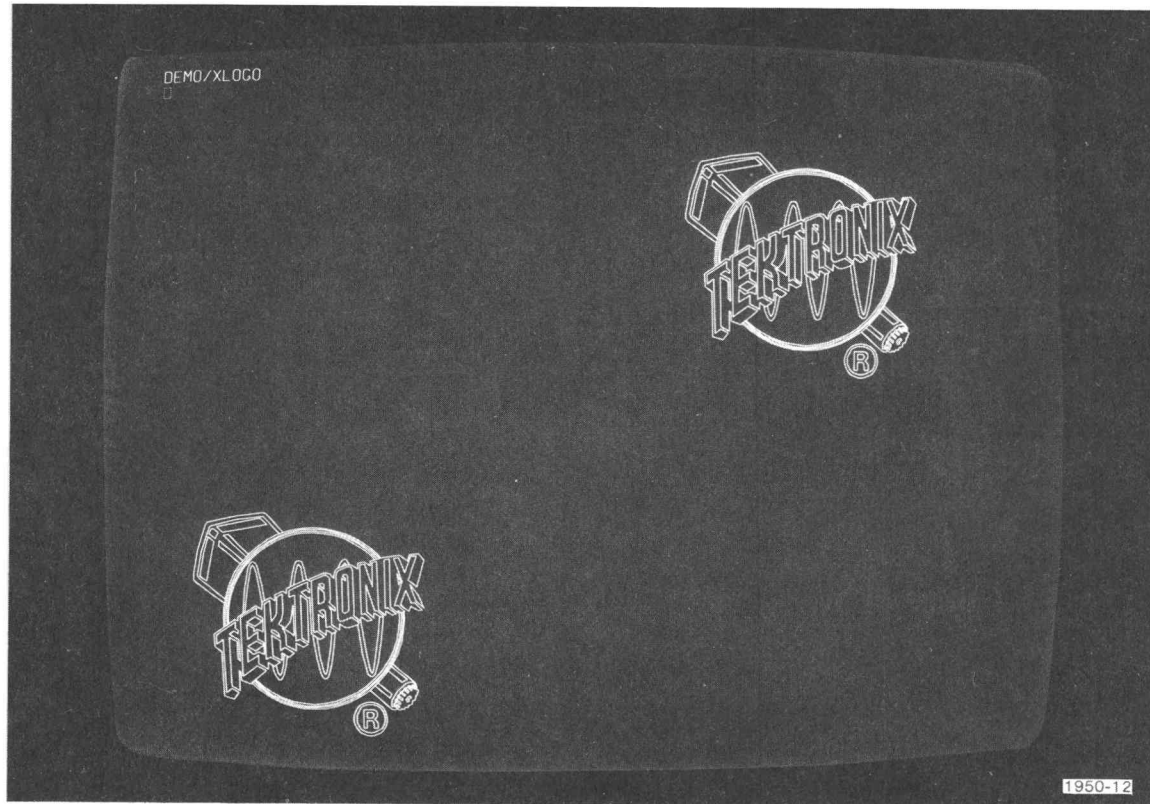


Fig. 6-2. Multiple Tektronix logos.

LANDER

The LANDER program demonstrates the concurrent use of a number of refresh objects. A lunar landscape, a fuel gauge, a timer, and a velocity component indicator are displayed on the Display Monitor in storage (Fig. 6-3). A lunar module, the amount of fuel available, the elapsed time, and the velocity components are displayed in refresh.

The object is to gently land the lunar module on the surface (the horizontal line above the fuel, time, and velocity component indicators). Gravity pulls the module toward the surface. A constant ether wind blows it slightly toward the right of the screen. To counteract these forces, the Joyswitch fires the lunar module's engines for braking or power.

To fire the braking engine, push the Joyswitch up. To fire the right or left power engine, move the Joyswitch in the direction the module is intended to go. The braking engine can be fired at the same time as the right or left engine by moving the Joyswitch diagonally upward. The Joyswitch also can be moved diagonally downward to employ either the right or left engine with gravity.

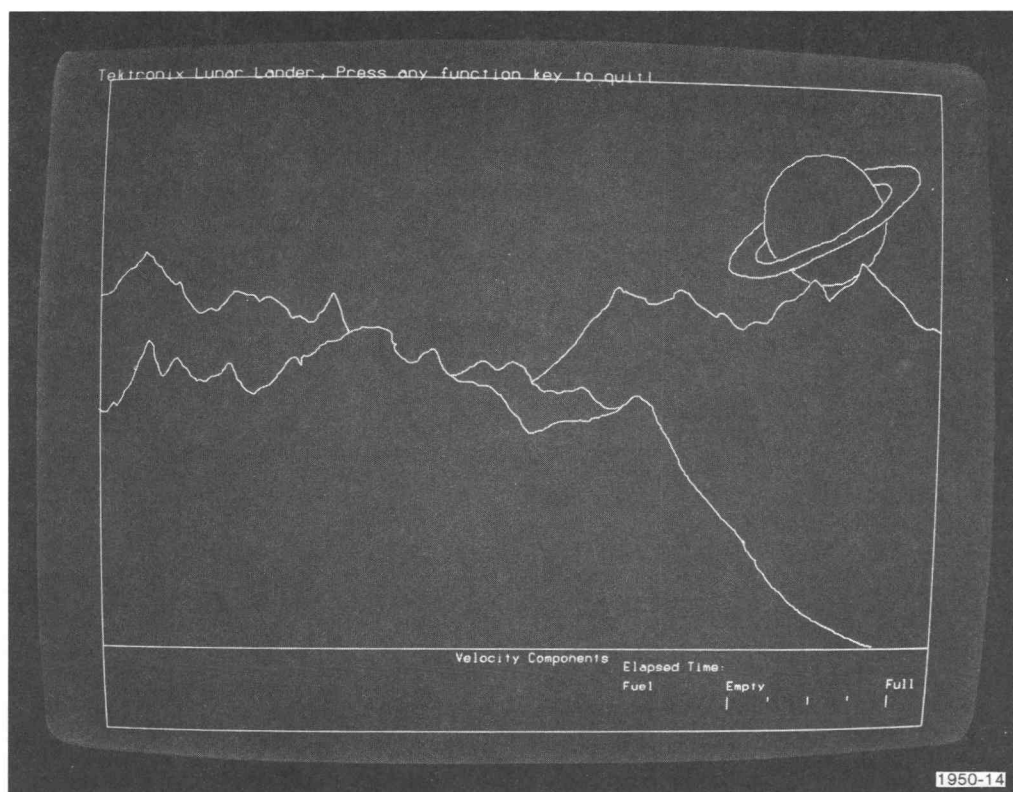


Fig. 6-3. LANDER lunar landscape and indicators.

Firing the engines consumes fuel. When 3/4 of the fuel is gone, the warning message => **LOW FUEL** <= appears on the screen and a warning beep is sounded.

As the module approaches the moon, lunar dust surges forth from the surface. If the approach is too fast, the warning message => **TOO FAST** <= appears on the screen.

When successfully landed on the surface, the module is displayed in storage at the landing location (Fig. 6-4). If the landing is unsuccessful, the screen is erased and the program is automatically restarted.

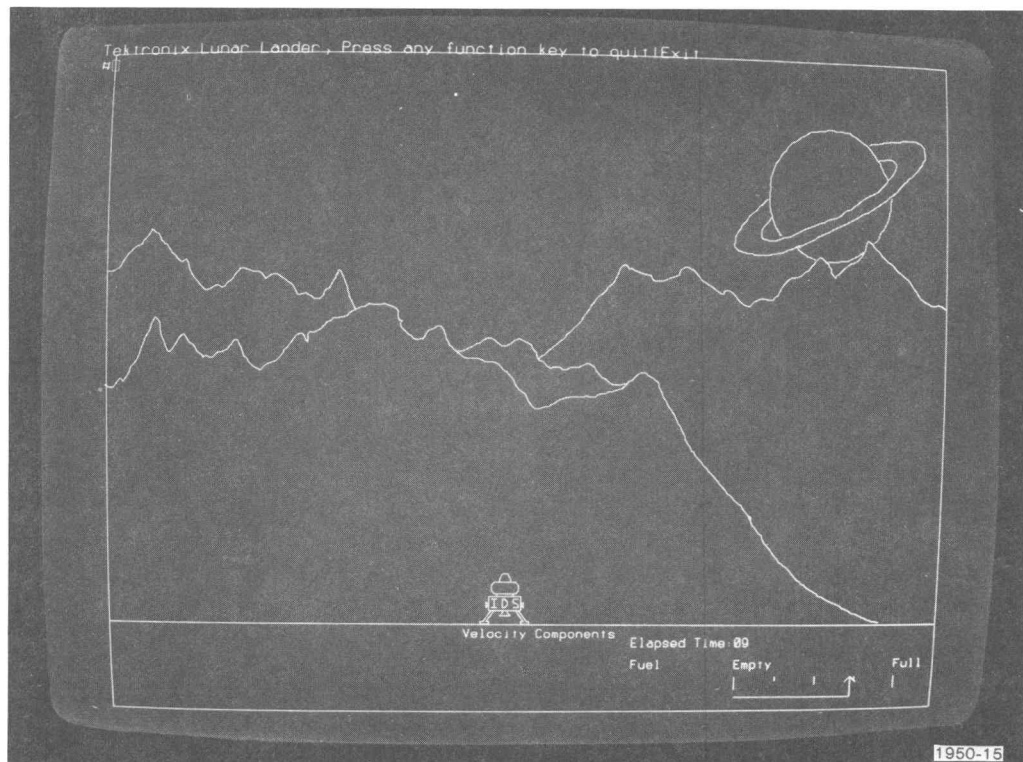


Fig. 6-4. Landed lunar module.

To run the LANDER program:

Type: **DEMO/LANDER**
Press: **RETURN**

Pressing any function key terminates the program and displays the following on the screen:

Exit
#

To restart the program after a successful landing, type **STA** and press **RETURN**. The module and the fuel, time, and velocity indicators from the previous landing remain on the screen unless the **RESET/PAGE** key is also pressed to erase the screen.

CUBE

The CUBE program demonstrates the use of the function keys to manipulate a three-dimensional object (a 12-sided convex polyhedron). To start the program:

Type: **DEMO/CUBE**
Press: **RETURN**

The function assigned to each function key is then displayed on the screen:

INIT XROT YROT ZROT HOLD PERS HLIN VPOUT VPIN STOP

To demonstrate three-dimensional manipulation, press the function keys in the order described in Table 6-1. Pause after pressing each function key to view the result. Table 6-2 explains the operation of each function key in greater detail.

Table 6-1
THREE-DIMENSIONAL MANIPULATION SEQUENCES

Function Key Sequences	Operation
0-1-2-3	Initializes the program (nothing appears on the screen) and rotates the object around each of the three axes.
5-6	Changes the perspective of the object and removes its hidden lines.
8-7-6-8	Slowly moves the operator's viewpoint toward the object, then away from the object, adds the hidden lines, then moves the viewpoint toward the object.
4-3-2-1	Holds the object in its current position, then rotates it around each of the three axes.
6-5-11	Removes the object's hidden lines, changes the perspective of the object, and terminates the program.

To restart the program:

Type: **DEMO/CUBE**
Press: **RETURN**

Table 6-2 lists the operations assigned to each function key. Pressing the keys in an order other than the sequences listed in Table 6-1 can produce strange and interesting results.

Table 6-2
FUNCTION KEY OPERATIONS

Function Key	Operation
0	Initializes the program.
1	Rotates the object around the x-axis.
2	Rotates the object around the y-axis.
3	Rotates the object around the z-axis.
4	Holds the object in its current position.
5	Changes the perspective of the object.
6	Removes/adds the object's hidden lines.
7	Slowly moves the operator's viewpoint away from the object.
8	Slowly moves the operator's viewpoint toward the object.
11	Terminates the program and displays the following: Exit #

HAND

The HAND program uses a graphic representation of a hand to demonstrate a non-standard graphic input cursor (the GOS standard graphic input cursor is the crosshair cursor). To display the hand:

Type:	DEMO/HAND
Press:	RETURN

The hand is displayed on the Display Monitor in refresh. The reference point of the cursor is initially set at the fingertip of the hand. To rotate the hand clockwise, depress Function Key 11. To rotate it counterclockwise, depress Function Key 10.

As the hand is moved across the screen under Joystick control, it draws a refresh line from the reference point to the current location of the fingertip. Depressing the SHIFT key while operating the Joystick increases the speed of the hand's movement.

Table 6-3 lists the commands that control the refresh line and the reference point. Each time one of the commands is typed, the reference point is moved to the current position of the fingertip.

Table 6-3
REFRESH LINE AND REFERENCE POINT MANIPULATION

Command	Operation
D (DRAW)	Draws the refresh line in storage.
M (MOVE)	Removes the refresh line.
E (ERASE)	Erases the screen (the hand reappears).

To draw the line in storage while the hand is moving, depress the D key while manipulating the hand with the Joyswitch. During this operation, the reference point moves with the fingertip.

NOTE

To use the SHIFT key with the D key-Joyswitch combination, depress the SHIFT key before adding the D key and the Joyswitch. The SHIFT key can be released after the D key-Joyswitch combination has been added without decreasing the speed of the hand.

Terminate the program by typing any alphanumeric character except one of the command characters (D, M, and E). The following appears on the screen:

Exit
#

To restart the program, type **STA** and press RETURN.

MOTOR

The MOTOR program demonstrates that seemingly difficult graphics applications can be made simple. It displays a 4-stroke, 2-speed, 1-cylinder motor on the Display Monitor. The motor's piston positions and valve cycles are completely table driven.

To run the program:

Type: **DEMO/MOTOR**
Press: RETURN

After an explanation of the program appears on the screen, type any keyboard character to begin the program (display the motor, label the function keys, and initiate motor activity). Fig 6-5 shows what is displayed on the screen.

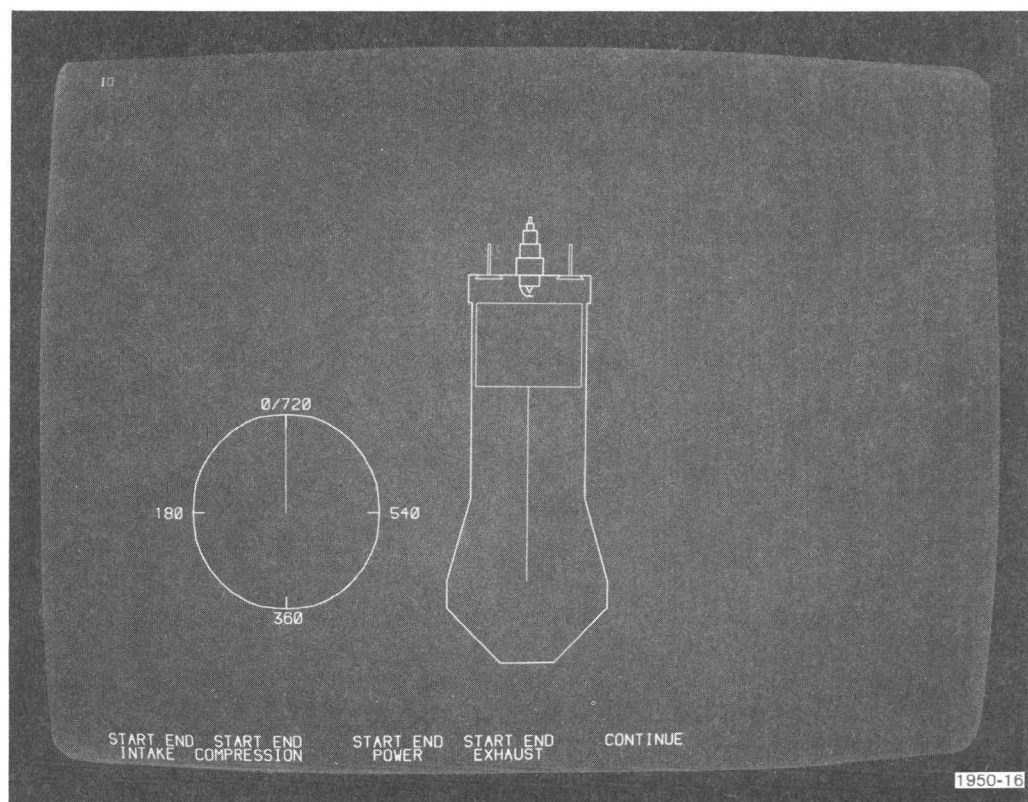


Fig. 6-5. MOTOR display.

The table values for the piston positions are shown in Table 6-4. The operator cannot change these values.

Table 6-4
PISTON POSITIONING

Piston Position	Degrees
Starting downward	0
Starting upward	180
Starting downward	360
Starting upward	540

Table 6-5 lists the commands that control the speed of the motor.

Table 6-5
MOTOR SPEED

Command	Operation
S (SLOW)	Determines low gear (10 1/3 rpm).
F (FAST)	Determines high gear (21 rpm).
I (INCREMENTAL)	Places the motor under Joyswitch control.

When the motor is in Incremental mode (after typing **I**), moving the Joyswitch in any direction rotates the motor slowly. The operator can change the table values for the valve cycles by stopping the motor (releasing the Joyswitch) at the desired points and pressing the appropriate function keys to alter the starting/ending points of subsequent cycles. Table 6-6 lists the functions assigned to the function keys. Each cycle can be modified ± 45 degrees from a perfect 180-degree cycle (the initial table values). Attempts to change cycle points to invalid positions are ignored.

Table 6-6
VALVE TIMING

Function Key	Operation	Display
0	Opens the left valve.	Writes INTAKE.
1	Closes the left valve.	Erases INTAKE.
2		Writes COMPRESSION.
3		Erases COMPRESSION.
4	Fires the ignition spark.	Writes POWER.
5		Erases POWER.
6	Opens the right valve.	Writes EXHAUST.
7	Closes the right valve.	Erases EXHAUST.
8	Exits Incremental mode.	

Typing the letter Q when the program is not in Incremental mode displays all refresh objects in storage and terminates the program. The following appears on the screen:

Exit
#

To restart the program, using the last values entered in the table for the valve cycles:

Type: **STA**
Press: **RETURN**

To restart the program, using the initial table values for the valve cycles:

Type: **DEMO/MOTOR**
Press: **RETURN**

LIFE

The LIFE program demonstrates the 4081 Graphic System's computing speed and its ability to perform complex calculations while displaying a number of refresh objects simultaneously.

LIFE graphically represents John Horton Conway's game of "life" by simulating the rise, fall, and survival of a society of living organisms. The 4081 representation of "life" is displayed on a 64 x 64 matrix with each cell value computed for each generation.

To run the program:

Type:	DEMO/LIFE
Press:	RETURN

A small triangle is displayed on the center of the screen in refresh. Manipulating the Joystick moves the triangle across the screen. To increase the speed of the triangle, depress the SHIFT key while operating the Joystick.

To set an organism in a cell, press the JOYSWITCH TERMINATOR when the triangle is in the desired location. This process can be repeated at various locations on the screen until the desired pattern is produced.

When the desired pattern of organisms has been produced, press any alphanumeric character followed by the JOYSWITCH TERMINATOR key. This sets an organism in the cell currently containing the triangle and initiates a graphic representation of the birth, death, and survival of the society of organisms according to the following genetic laws:

- Births: Each empty cell with only three neighbors (organisms in adjacent cells) will receive an organism in the next generation.
- Deaths: Each organism with four or more neighbors dies (from overpopulation) as does each organism with one or no neighbors (from isolation).
- Survivals: Each organism with two or three neighbors survives to the next generation.

All births and deaths occur simultaneously, without influencing each other, during the same generation. The population may undergo numerous changes before dying out or stabilizing into still or oscillating patterns. The following example (Fig. 6-6) shows the life history of a simple pattern (..) that undergoes eight changes before settling into the first oscillation of a stable oscillating pattern.

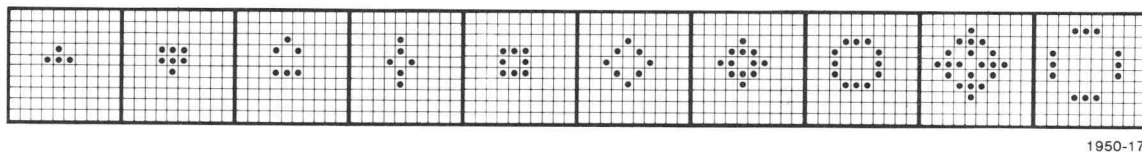


Fig. 6-6. Life history of a simple pattern.

Fig. 6-7 pictures the life history of a Cheshire cat pattern. Only a grin is left in the seventh generation. In the eighth generation, the grin has disappeared and only a paw print remains.

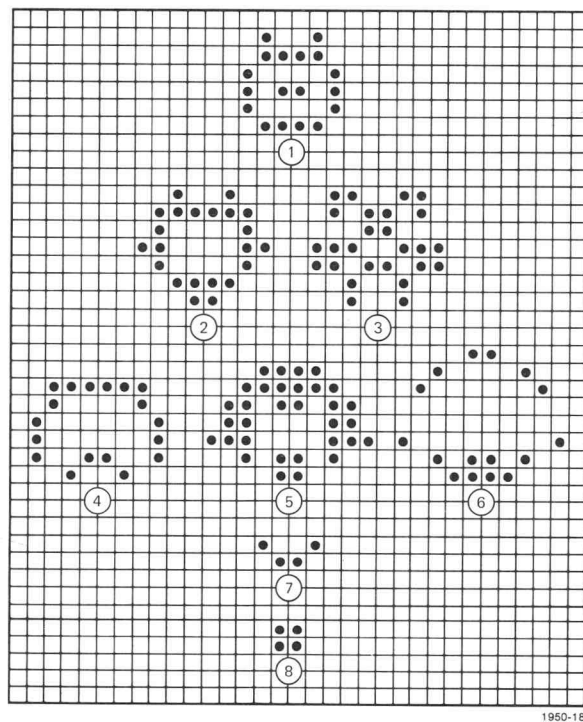


Fig. 6-7. Life history of a Cheshire cat pattern.

To terminate the program:

Depress the SHIFT key and press ESC

Type: **CLO**

Press: RETURN

The following appears on the screen:

Exit

#

To restart the program, type **STA** and press RETURN.



SECTION 7

SWITCHES, MESSAGES, MODES, LIGHTS, AND KEYS

CONTENTS

	Page
Power Switches	7-3
Initial Program Load (IPL)	7-5
GOS Standard Messages	7-7
Prompt Characters	7-8
MVP Full Condition	7-8
Type-Ahead Feature	7-8
Storage Display Monitor Operations	7-9
MONITOR ERASE	7-10
STORAGE HARD COPY	7-10
VIEW	7-10
Operating Modes	7-11
Command Mode	7-11
Program Mode	7-12
Host Mode	7-13
Keyboard Prompt Lights	7-15
Keyboard Operations	7-16
BREAK	7-18
CTRL	7-19
CTRL-N	7-19
CTRL-O	7-20
CTRL-R	7-21
CTRL-S	7-21
ESC (Escape)	7-23
LF (Line Feed)	7-23
RESET/PAGE	7-24
RETURN	7-24
RUBOUT	7-24
SHIFT	7-25
SHIFT-BREAK	7-25
SHIFT-ESC	7-25
SHIFT-RESET	7-26
SHIFT-RETURN	7-27
TAB	7-27
TTY LOCK	7-27

Section 7

SWITCHES, MESSAGES, MODES, LIGHTS, AND KEYS

POWER SWITCHES

The POWER switch for the Storage Display Monitor is on the lower section of the rear panel of the unit (Fig. 7-1). This switch is for service purposes only and should be left on at all times.



Fig. 7-1. Storage Display Monitor POWER switch.

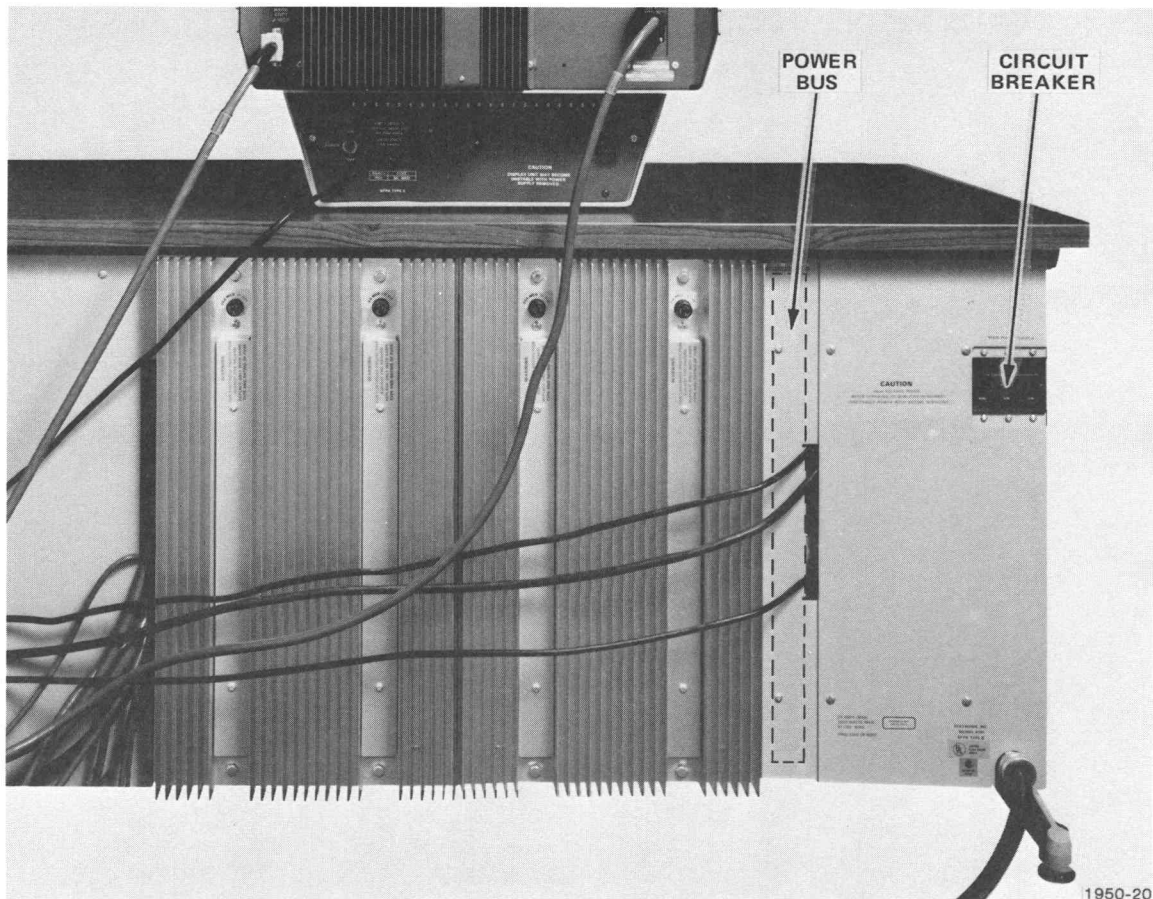


Fig. 7-2. System power bus and circuit breaker.

Although power to individual peripheral devices can be controlled by the power switch on each device, the preferred system installation enables all peripheral devices to obtain power from the 4081 Graphic System power bus (Fig. 7-2).

When the circuit breaker on the system power bus is on (up) (Fig. 7-2), power is applied to all system devices simultaneously by turning on the POWER switch located on the front of the left cabinet (Fig. 7-3). The power switches on the devices connected to the 4081 should remain on at all times. Use the 4081's front-panel POWER switch to turn the entire system on and off. Note that the green POWER light on the front of the Storage Display Monitor (Fig. 7-3) is lit when power is being supplied to the system.

INITIAL PROGRAM LOAD (IPL)

After power is applied to the 4081 Graphic System (see **Power Switches** in this section) and all peripheral devices are on and ready, the Graphic Operating System (GOS) program must be loaded into the 4081 memory. The load procedure is known as Initial Program Load (IPL). The GOS program must be loaded into memory using the IPL procedure every time the system is turned on.

To initiate the IPL procedure immediately after turning on the power, insert the GOS IPL tape cartridge in the Cartridge Tape Unit drive assigned the device mnemonic CTO: (generally, the top drive unless the device addresses of the Cartridge Tape Unit drives have been changed by hardware modification). The BUSY light, on the upper right of the Cartridge Tape Unit (Fig. 7-3), is lit whenever the tape is moving.

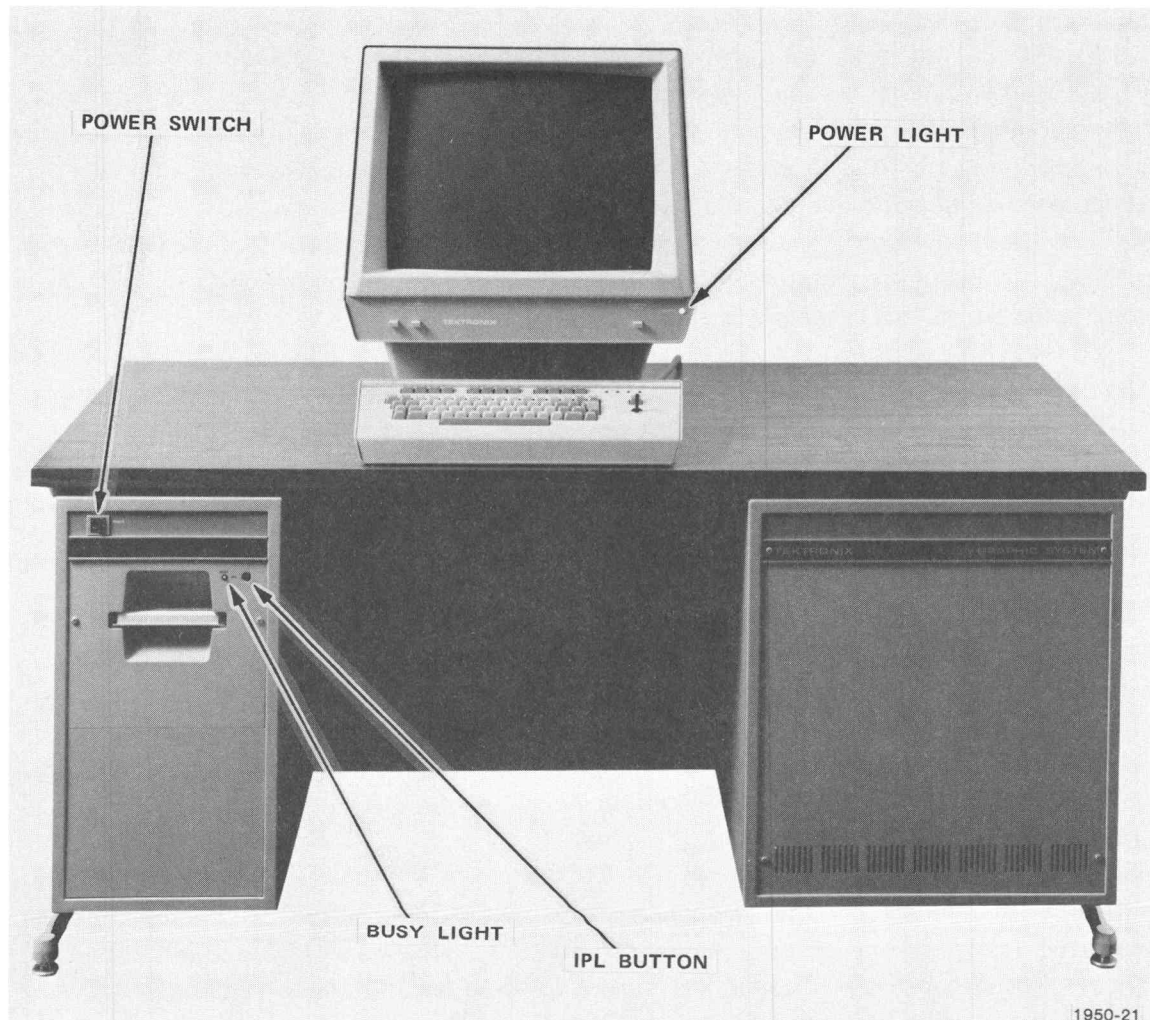


Fig. 7-3. 4081 POWER switch, POWER light, BUSY light, and IPL button.

SWITCHES, MESSAGES, MODES, LIGHTS, AND KEYS

When GOS has been loaded into the 4081 memory (10 to 15 seconds), the display screen is erased, a blinking alphanumeric cursor appears, and the GOS IPL tape cartridge rewinds. If this sequence does not occur, be sure the correct GOS IPL tape cartridge is inserted properly in the drive, then press the IPL button (Fig. 7-3).

After the GOS IPL tape cartridge rewinds, a GOS identification message appears in the following form on the upper left of the display screen:

GOS Vx.xLx
#

Vx.x is the version number and **Lx** is the level number of the GOS program. After the message is displayed, the keyboard bell rings.

On some 4081's the operator may notice additional information displayed with the GOS version and level numbers and the **#** prompt character. This additional information results from a convenient feature available as a GOS program option. If this option is selected, then as soon as the GOS program is loaded into the 4081 memory, GOS automatically executes a sequence of commands that have been loaded into memory as if typed on the keyboard. In most cases, the only command loaded into memory as part of the GOS IPL procedure is the utility program BATCH. The BATCH program causes a sequence of commands stored in a batch file to be executed on the 4081.

The batch file is a file on the system disc (or tape). The commands stored in the batch file are selected by the user. The commands can be any combination of GOS resident commands, GOS utility programs, and user programs. The Tektronix field representative, when creating the user's GOS IPL tape cartridge, includes a command to GOS to run the BATCH program and execute the user's batch file. This option is available as part of the standard 4081 Graphic System.

The user's batch file generally contains the commands most often typed by an operator after the 4081 is turned on and the GOS program is loaded into memory. By including the commands in the batch file, GOS executes the commands automatically, saving the operator the trouble of typing them. For example, if a 4081 is used mainly to communicate with a host computer, the user might include a command in the batch file to initialize the primary communications interface (SET COM).

If the error message **Dir I/O error DU lu 16** is displayed on the screen, GOS could not run the BATCH program and execute the user's batch file. In this case, make sure that the system disc (or tape) is inserted in the correct device drive, that the drive door is shut, and that the device is on and properly connected to the 4081.

After the GOS program is loaded into memory, remove the IPL tape cartridge, store it in a safe place, and insert the system tape or disc in the appropriate device drive. (For information on inserting a tape cartridge, flexible disc, or hard disc, see **File-Structured Devices** in the section **Files**). Now the operator can run any resident command or utility program on the system device.

Any time the system power is on, the operator can reload GOS by inserting the GOS IPL tape cartridge and pressing the IPL button. Since the current contents of the 4081 memory are destroyed when the GOS program is loaded into memory, do not initiate the IPL procedure unless the information stored in memory is unnecessary or stored elsewhere (for example, on a flexible disc or tape cartridge).

In summary, the following steps should be taken to ready the 4081 Graphic System for operation after it has been installed and checked out by Tektronix personnel.

1. Turn on the system power and make sure that all peripheral devices are ready.
2. Insert the GOS IPL tape cartridge in the correct Cartridge Tape Unit drive (generally, the top drive).
3. Wait for the GOS identification message. If the message does not appear within 20 seconds, check the tape and press the IPL button.
4. Remove the GOS IPL tape cartridge and store it in a safe place.
5. Insert the system tape or disc in the appropriate device drive.

GOS STANDARD MESSAGES

All system identification messages, error messages, displayable operator input, and selected program output are displayed on the 4081's Storage Display Monitor. The information is displayed on a portion of the screen called the monitor viewport (MVP). Unless modified by the SET utility program (see **SET** in the section **GOS Utility Programs**), the MVP is the entire visible screen.

Output to the MVP begins at the upper left corner, with subsequent lines of text displayed down the screen. The RESET/PAGE key erases the display and moves the alphanumeric cursor to the "home" position (the upper left corner of the MVP). (For a list of GOS standard messages, see the appendix **General System Messages**.)

PROMPT CHARACTERS

There are three GOS standard prompt characters that are displayed on the monitor viewport (MVP) to request keyboard input from the 4081 Graphic System operator: #, \$, and *.

#

Requests the name of a GOS resident command to be executed or a GOS utility program or user program to be run.

,

General purpose input requests. The input depends on the program that is running on the 4081. The input is generally a filename or other argument for the program.

MVP FULL CONDITION

If all the available lines in the monitor viewport (MVP) have been used (when the 4081 is not operating in Host mode), the MVP FULL condition occurs. The alphanumeric cursor is automatically moved to the "home" position, the keyboard prompt light labeled MVP FULL is lit, and output to the MVP (other than the display of characters typed on the keyboard) is prevented until the MVP FULL condition is cleared.

Clearing the MVP FULL condition allows the running program to accept typed-ahead input and to display further information on the Display Monitor. Pressing the RESET/PAGE key clears the MVP FULL condition and erases the screen. Typing the letter R while depressing the CTRL key also clears the MVP FULL condition but does not erase the screen. (For further information on the RESET/PAGE and CTRL-R functions, see **Keyboard Operations** in this section.)

TYPE-AHEAD FEATURE

The type-ahead feature allows the operator to type up to 130 characters while the MVP FULL condition is in effect. The characters are displayed on the screen as they are entered. The typed-ahead input can include multiple commands and programs, each followed by pressing the RETURN key or the LF key. To transmit all of the typed-ahead characters for processing after the MVP FULL condition is cleared, the last key pressed must be either the RETURN key or the LF key.

A bell rings if the operator attempts to type more than 130 characters while the MVP FULL condition is in effect. Pressing the RUBOUT key deletes the 130th character so that the operator can enter the RETURN character or the LF character. Clearing the MVP FULL condition begins the execution of the typed-ahead commands and programs.

The type-ahead feature, for example, allows an operator to type a sequence of commands and programs while the MVP FULL condition is in effect. Then, after clearing the MVP FULL condition, the operator can leave the 4081 while the typed-ahead commands and programs are executed.

(To prevent the occurrence of the MVP FULL condition, set the internal Job Status Word, the JSW, as instructed in the Plot 80: GOS Programmer's Reference manual.)

STORAGE DISPLAY MONITOR OPERATIONS

Three buttons are located on the front panel of the Storage Display Monitor: the MONITOR ERASE button, the STORAGE HARD COPY button, and the VIEW button (Fig. 7-4).

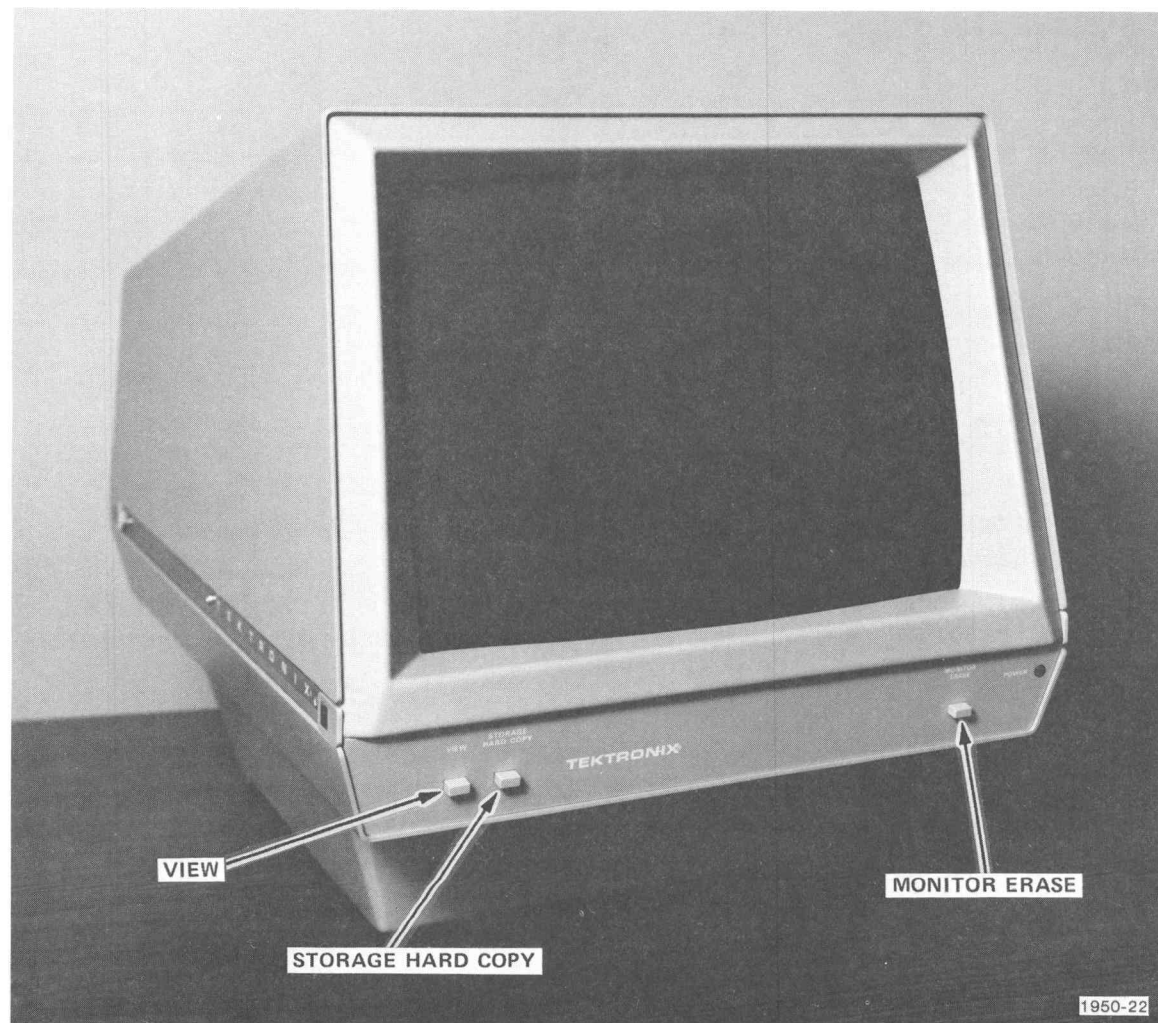


Fig. 7-4. MONITOR ERASE, STORAGE HARD COPY, and VIEW buttons.

MONITOR ERASE

Pressing the MONITOR ERASE button erases the screen without repositioning the alphanumeric cursor and without clearing the MVP FULL condition. When GOS is running on the 4081, however, it is recommended that the operator erase the screen by pressing the RESET/PAGE key. In addition to erasing the screen, the RESET/PAGE key moves the alphanumeric cursor to the upper left corner of the monitor viewport (MVP) and clears the MVP FULL condition.

STORAGE HARD COPY

If the user's 4081 Graphic System includes a hard copy device, pressing the STORAGE HARD COPY button produces a copy of the stored image displayed on the Display Monitor. During the copy cycle (while the hard copy scanning bar sweeps the screen), no further information can be displayed on the screen.

VIEW

If an image is displayed on the screen for approximately 90 seconds without further output to the Display Monitor, Hold state takes effect (the screen becomes darker and the intensity of the screen image is dimmed). Pressing the VIEW button restores the Display Monitor to the View state. Pressing the SHIFT key or any other key also restores the Display Monitor to the View state.

CAUTION

To maximize the life of the Display Monitor, the same information should not remain in View state for more than 15 minutes or in Hold state for more than one hour. Press the MONITOR ERASE button immediately to erase the screen if any of the following events occur:

- *The IPL procedure is not initiated immediately after power is applied to the 4081 Graphic System.*
- *The 4081 is not used immediately after the GOS identification message appears when the IPL procedure is completed.*
- *The system becomes inoperative with data displayed on the screen.*

Residual images that remain after the screen is erased usually can be removed by erasing the screen several times.

OPERATING MODES

Generally, an operator communicates information to the 4081 Graphic System by typing the information on the keyboard. The information, however, can be sent to any one of three destinations depending on the mode in which the 4081 is functioning. The three modes are:

- Command mode
- Program mode
- Host mode

Command Mode

When the 4081 is functioning in Command mode, all characters typed on the keyboard, except characters assigned special functions by GOS (for example, the ESC and RUBOUT keys), are sent to the GOS command processor when the RETURN or LF key is pressed. If 80 characters are typed before pressing the RETURN or LF key, a line feed is automatically inserted and the 80 characters are sent to the command processor.

The command processor interprets all keyboard data it receives as a resident command or a program file to be executed. Generally, if the keyboard data is not a valid resident command or program file, an error message is displayed.

While the 4081 is in Command mode, any typing errors can be corrected using the RUBOUT and ESC keys before sending the characters to the command processor. RUBOUT deletes the last character typed. ESC deletes the entire line.

To enter Command mode from Host mode or Local mode, press the SHIFT key and, while depressing the SHIFT key, press the ESC key. The message <Attn> is displayed followed, on the next line, by the prompt character #. The prompt character # indicates that the command processor is ready to receive keyboard data. The 4081 operates initially in Command mode after the GOS program is loaded into memory.

Program Mode

When the 4081 is functioning in Program (or Local) mode and a program is running on the 4081, the program determines the destination of all characters typed on the keyboard, except characters assigned special functions by GOS.

A program can be written to accept one keyboard character at a time, a line of keyboard characters at a time, or no keyboard characters. Generally, when a program is ready to receive characters from the keyboard, it displays a prompt on the screen. A prompt can be a phrase (for example, "Pick a filename, any filename") or a single character (for example, "?" or "\$").

If the program accepts one character at a time, the character is sent to the program as soon as it is typed. In this case, the character cannot be edited (deleted) if an error is made.

If the program accepts a line of keyboard characters at a time, the characters are sent to the program when either the RETURN or LF key is pressed, or when 80 characters are typed before the RETURN or LF key is pressed. Usually, any typing errors can be corrected using the RUBOUT and ESC keys before sending the characters to the program. RUBOUT deletes the last character typed. ESC deletes the entire line. GOS, however, gives the program the option of treating the RUBOUT and ESC keys as ASCII characters with no editing functions. In this case, pressing the RUBOUT key displays a crosshaired rectangle on the screen in the next character position and pressing the ESC key displays Λ [. In ASCII, the ESC key is equivalent to pressing the [(left square bracket) key while depressing the CTRL key.

If the program is not written to accept any characters, all characters typed on the keyboard, except characters assigned special functions by GOS, are discarded.

The 4081 enters Program mode when a program is run and remains in Program mode until the program pauses, exits, or is terminated by GOS because of an error. Program mode can be interrupted by pressing the SHIFT-ESC keys to enter Command mode or by pressing the SHIFT-BREAK keys to enter Host mode.

When Program mode is interrupted to enter Command mode, the running program is stopped at the point of interruption. If no other programs are loaded into memory and executed while in Command mode, then the program can be continued from the point of interruption by typing CON and pressing the RETURN key.

When Program mode is interrupted to enter Host mode, the program still continues to run until it ends. The program can still display information on the screen and any GOS error messages relating to the program are also displayed. While in Host mode, however, the program cannot receive characters from the keyboard (all characters are sent through the primary communications interface).

If the program running on the 4081 pauses or exits while in Host mode, the 4081 automatically enters Command mode. In this case, the message **Pause** or **Exit** is displayed followed, on the next line, by the prompt character **#**. The prompt character **#** indicates that the command processor is ready to receive keyboard data. The operator can return to Host mode by pressing the SHIFT-BREAK keys.

To reenter Program mode from Host mode, press the SHIFT key and, while depressing the SHIFT key, press the RETURN key. The message **<Local>** is displayed on the screen. When the 4081 is in Command mode, however, pressing the SHIFT-RETURN keys has the same result as pressing only the RETURN key.

Host Mode

When the 4081 is functioning in Host mode, all characters typed on the keyboard are sent through the primary communications interface. Generally, the primary communications interface is connected to a host computer, but it can also be connected to an external device (like a plotter or line printer).

Before characters can be transmitted in Host mode through the primary communications interface, the operator must initialize the primary communications interface. Initializing the primary communications interface includes assigning it a set of data communications options (echo, parity, speeds, number of stop bits, and baud rate) that determine how the data is transmitted and received. The operator initializes the primary communications interface using the GOS utility program SET while the 4081 is in Command mode. (See the section **Data Communications In Host Mode** for more detailed information.)

In Host mode, the 4081 keyboard functions as a full-ASCII alphanumeric terminal, similar to a teletypewriter. Each character typed on the keyboard is sent directly to the host computer. In case of typing errors, the characters cannot be edited (deleted) by the 4081 before they are transmitted. The host computer, however, usually provides the capability to edit the characters it receives.

To enter Host mode, press the SHIFT key and, while depressing the SHIFT key, press the BREAK key. The message **<Host>** is displayed on the screen.

When the 4081 enters Host mode from Program mode, the program running on the 4081 continues to run as normal except that it cannot receive characters from the keyboard. Both the running program and the host computer can display information on the 4081 Display Monitor. The running program can also send information through any of the communications interfaces (there are a maximum number of six communications interfaces) including the primary communications interface.

SWITCHES, MESSAGES, MODES, LIGHTS, AND KEYS

If the running program pauses or exits while in Host mode, the 4081 automatically enters Command mode. In this case, the message **Pause** or **Exit** is displayed followed, on the next line, by the prompt character **#**. The operator can return to Host mode by pressing the SHIFT-BREAK keys.

While in Host mode, if the operator presses the SHIFT-ESC keys, the 4081 enters Command mode. The message **<Attn>** is displayed followed, on the next line, by the prompt character **#**.

While in Host mode, if the operator presses the SHIFT-RETURN keys, the 4081 enters Program mode if a program is running on the 4081 or Command mode if no program is running. The message **<Local>** is displayed on the screen.

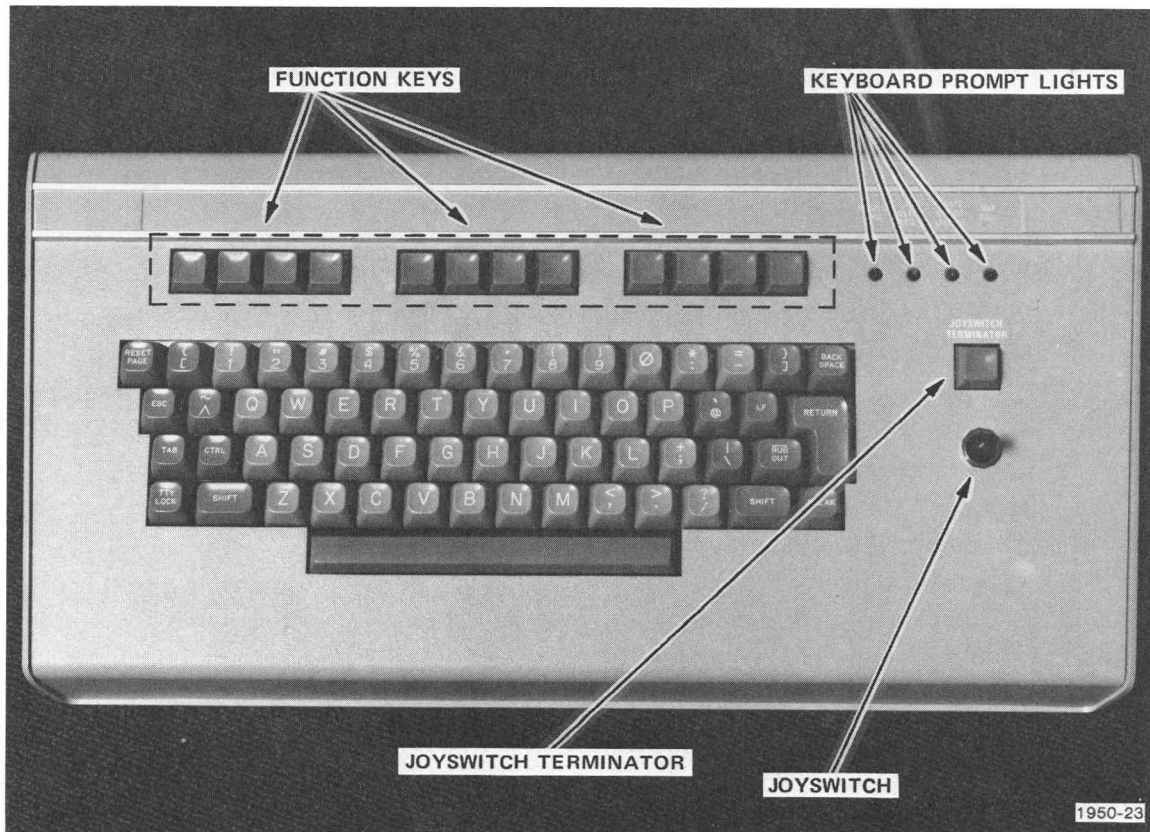


Fig. 7-5. Keyboard prompt lights, Joystick, function keys, and JOYSWITCH TERMINATOR key.

KEYBOARD PROMPT LIGHTS

Fig. 7-5 shows the four red prompt lights located on the upper right of the Keyboard Unit. When the 4081 Graphic System is not operating in Host mode, the keyboard prompt lights generally indicate the status of the keyboard and the monitor viewport (MVP). The prompt lights are programmable, however, and can vary from program to program (for example, the PLOT 80:4014 Emulator program). Consult the documentation for the program running on the 4081 for the current significance of the prompt lights.

Generally, the keyboard prompt lights are lit to indicate the following conditions:

REQUEST INPUT	The REQUEST INPUT light indicates that input from the keyboard is needed. The input must be terminated by pressing the RETURN key or the LF key.
STOP OUTPUT (CTRL-S)	The STOP OUTPUT light indicates that the operator has suppressed output directed to the Display Monitor by depressing the CTRL key and typing the letter S. Entering a subsequent CTRL-S displays the suppressed output.
DELETE OUTPUT (CTRL-O)	The DELETE OUTPUT light indicates that the operator has discarded output to the Display Monitor by depressing the CTRL key and typing the letter O. This condition is cleared when a subsequent CTRL-O is entered, when additional keyboard input is requested, or when a running program clears the condition.
MVP FULL	The MVP FULL light indicates that all the available lines in the MVP have been used and the alphanumeric cursor has been moved to the "home" position (the upper left corner of the MVP). (For information on clearing the MVP FULL condition, see MVP FULL Condition in this section.)

KEYBOARD OPERATIONS

CAUTION

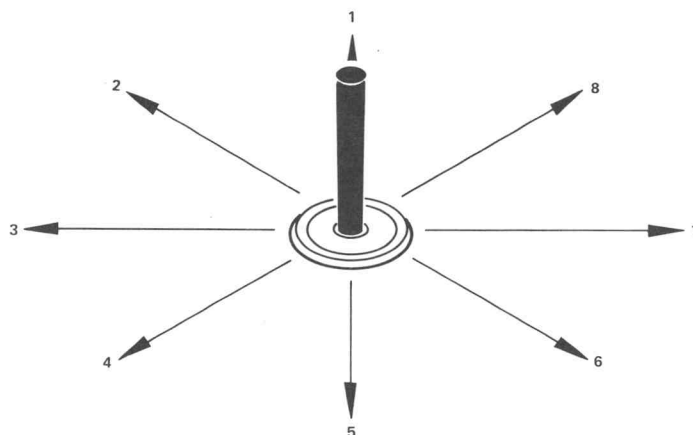
Avoid spilling liquids on the Keyboard Unit.

Keyboard operation is controlled by the Graphic Operating System (GOS) or the application program that is running on the 4081. GOS allows the operator to communicate with the 4081 Graphic System by typing information on the keyboard. The keys on the Keyboard Unit (Fig. 7-6) correspond to alphanumeric characters that conform to the American Standard Code for Information Interchange (see the appendix **ASCII Code Chart**).

The Keyboard Unit also contains a set of 12 function keys, a Joyswitch, and a JOYSWITCH TERMINATOR key (Fig. 7-5). The function keys can be assigned functions by a program to speed the input of commands and provide special operating procedures. Programs that assign functions to the function keys generally have overlay strips that label the specific function assigned to each key. (Refer to the Plot 80: GOS Programmer's Reference manual for information on programming the function keys and the Joyswitch.)



Fig. 7-6. Keyboard.



1950-25

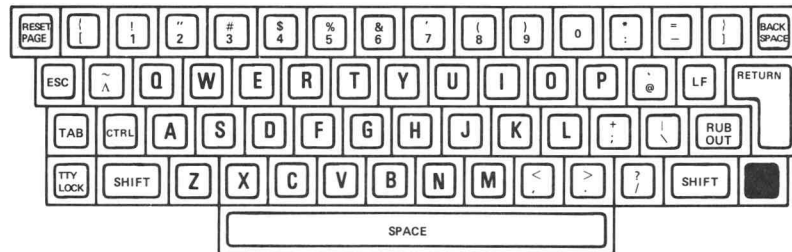
Fig. 7-7. The eight positions of the Joystick.

Fig. 7-7 shows the eight possible positions of the Joystick. The Joystick is generally used to position a graphic input cursor displayed on the 4081 Display Monitor. Pressing the JOY SWITCH TERMINATOR key (located directly above the Joystick) tells the 4081 Graphic System that a Joystick operation has been completed (similar to pressing the RETURN key to indicate completion of keyboard input).

The following section explains the operation of keys assigned special functions by GOS. It also includes examples that demonstrate the use of these keys. The characters that are printed in boldface type, in most cases, represent the information that is actually displayed on the screen.

NOTE

When two keys separated by a hyphen are specified (for example, CTRL-R), the second key must be pressed while depressing the first key.

BREAK

The BREAK key, when typed after pressing the RETURN key or the LF key, serves as an end-of-file character for entering keyboard input into a file.

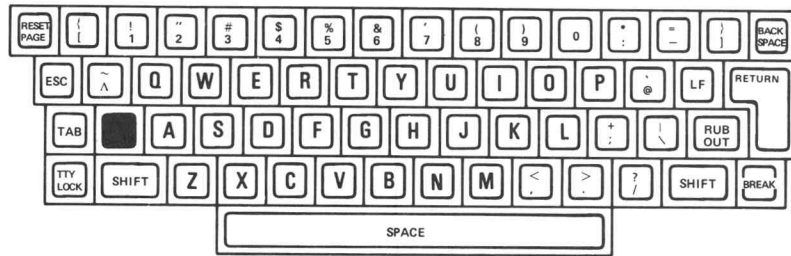
Example

4081 displays:	#
Type:	COPY TEST.ASM=KB:
Press:	RETURN
4081 displays:	-Files copied-
Type data for file named TEST.ASM:	SAY HELLO TO YOUR COMPUTER.
Press:	RETURN
Type more data for TEST.ASM:	HELLO, 4081.
Press:	RETURN
Press:	BREAK (to indicate end-of-file)
4081 displays:	#

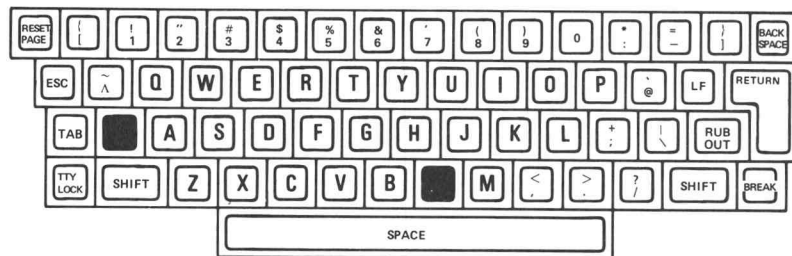
The file TEST.ASM now contains the following:

SAY HELLO TO YOUR COMPUTER.
HELLO, 4081.

The BREAK key is also used with the SHIFT key to enter Host mode (see the description of the SHIFT-BREAK keys).

CTRL

The CTRL key has no independent function under GOS but is used with the N, O, R, and S keys.

CTRL-N

When the MVP FULL condition is in effect (and only when the MVP FULL condition is in effect), typing the letter N while depressing the CTRL key moves the alphanumeric cursor to the next column on the screen without erasing the screen. There are eight columns preset on the screen. The columns remain the same regardless of the Display Monitor character size. When used with CTRL-R, CTRL-N allows the display of information in columns on the screen, increasing the amount of visible data.

Example

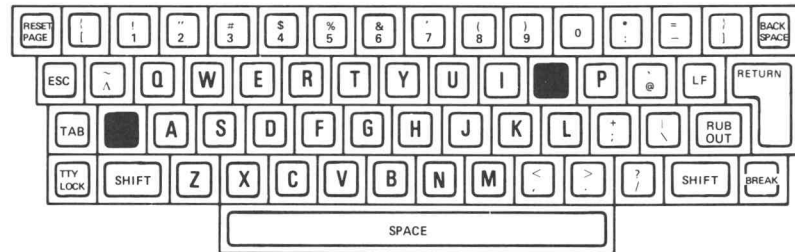
The MVP FULL condition occurs while data is being displayed on the screen.

Press CTRL-N.

The cursor moves to a new column on the screen.

Press CTRL-R

The MVP FULL condition is cleared without erasing the information previously displayed on the screen. Information now continues to be displayed on the screen (in a new column).

CTRL-O

Typing the letter O while depressing the CTRL key prevents information that would otherwise be displayed on the screen from being displayed. The CTRL-O function can save time by eliminating the display of unnecessary information.

Example

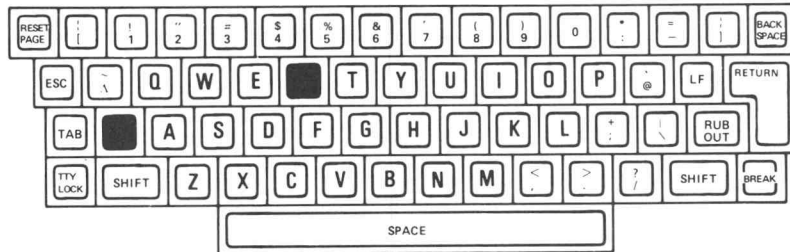
The operator wishes to see the number of blocks a specific file uses.

As the directory is being displayed on the screen, the operator presses CTRL-O after the desired information is displayed. The remainder of the directory is not displayed.

The DELETE OUTPUT prompt light is lit while the CTRL-O condition is in effect. The condition continues until additional keyboard input is requested, until another CTRL-O is typed, or until a running program clears the condition.

Example

If a file is long enough, the first part of the file can be displayed before typing CTRL-O. Then, after typing CTRL-O to delete the display of the middle of the file, the operator can type CTRL-O again to display the end of the file.

CTRL-R

Typing the letter R while depressing the CTRL key clears the MVP FULL condition without erasing the screen. When used with CTRL-N, CTRL-R allows the display of information in columns on the screen.

Example

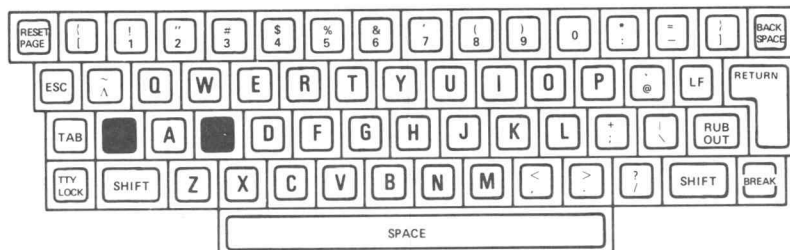
The MVP FULL condition occurs while data is being displayed on the screen.

Press CTRL-N.

The cursor moves to a new column on the screen.

Press CTRL-R.

The MVP FULL condition is cleared without erasing the information previously displayed on the screen. Information now continues to be displayed on the screen (in a new column).

CTRL-S

Typing the letter S while depressing the CTRL key suspends the display of information on the screen. Characters can still be typed on the keyboard, but they are not displayed on the screen. The STOP OUTPUT prompt light is lit when the CTRL-S condition is in effect. When the CTRL-S condition is cleared, by pressing the CTRL-S keys again, the first 25 characters typed on the keyboard during the CTRL-S condition are displayed.

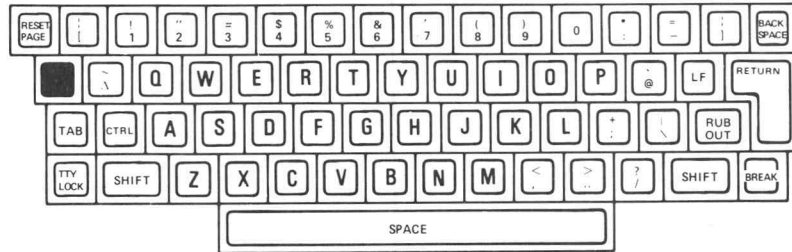
If the operator runs a program or a program is running on the 4081 when the CTRL-S condition is in effect, the program continues to run unless it is waiting for the completion of a display operation. Clearing the CTRL-S condition by pressing the CTRL-S keys again allows a waiting program to complete the display operation and continue.

Since many GOS resident commands, GOS utility programs, and user programs involve some form of display operation (from displaying a message on the screen to a picture), the CTRL-S condition allows the operator to enter a sequence of commands and programs for execution. The operator must press the RETURN key or the LF key after typing each command or program name in the sequence. The total number of characters typed must not exceed 80 (130 if the MVP FULL condition is in effect). When the CTRL-S condition is cleared, the commands and programs are executed in the order in which they were typed.

Example

An animated picture is displayed on the screen in refresh. The animated picture shows a rocket moving in a parabolic pattern like a rainbow across the screen. A counter on the bottom of the screen continually computes and displays the rocket's height and speed at certain points in its trajectory. To stop the movement of the rocket and check its height and speed, the operator presses the CTRL-S keys. Since the CTRL-S condition suspends further display operations, the rocket appears frozen on the screen. To continue the rocket's movement along its trajectory, clear the CTRL-S condition by again pressing the CTRL-S keys.

ESC (Escape)

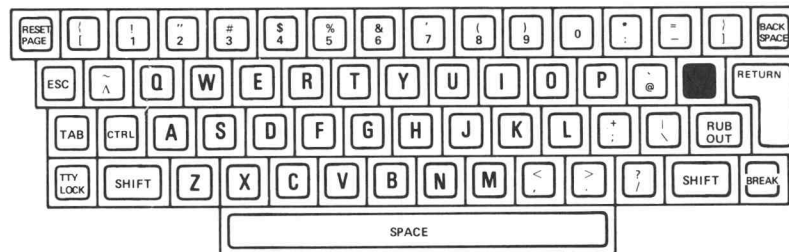


The ESCAPE key deletes the current line of keyboard input, displays **** after the deleted line, and moves the alphanumeric cursor to the beginning of the next line. It is also used with the SHIFT key to enter Command mode (see the description of the SHIFT-ESC keys).

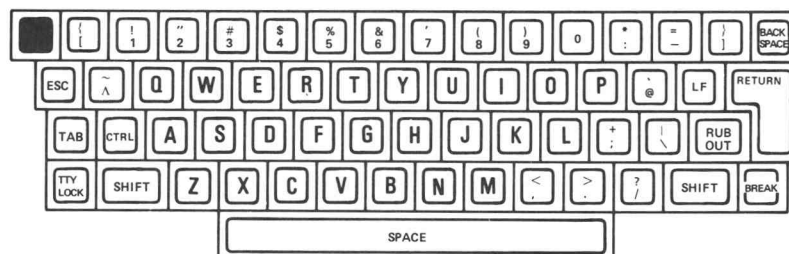
Example

```
#*THIS IS A COMMA <Del>
*THIS IS A COMMENT
```

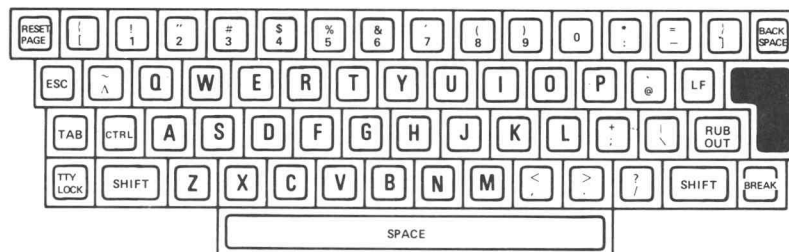
LF (Line Feed)



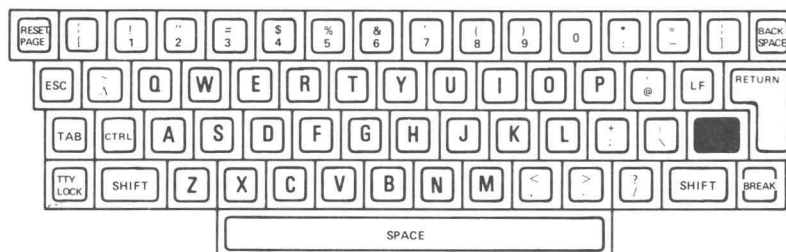
The LF key transmits a line of keyboard input for processing (to the GOS command processor or to a running program) without moving the alphanumeric cursor to the next line. The LF key can also be assigned special functions by a program running on the 4081 (for example, the Plot 80: GOS TECO program).


RESET/PAGE

The RESET/PAGE key erases the display and moves the alphanumeric cursor to the “home” position (the upper left corner of the MVP). The RESET/PAGE key also clears the MVP FULL condition.

RETURN

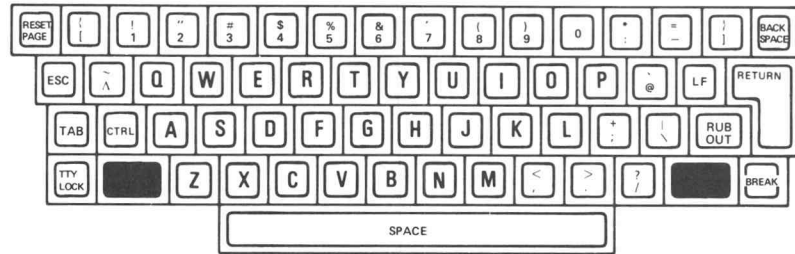
The RETURN key transmits a line of keyboard input for processing (to the GOS command processor or to a running program) and moves the alphanumeric cursor to the beginning of the next line. It also is used with the SHIFT key to enter Program mode (see the description of the SHIFT-RETURN keys).

RUBOUT

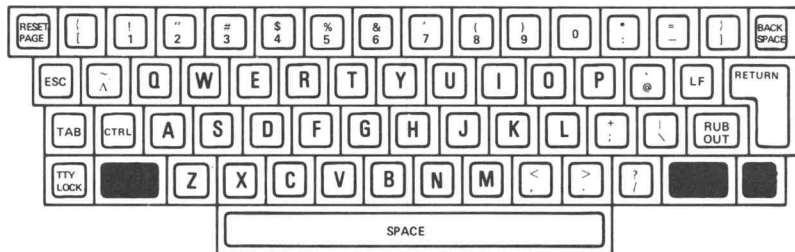
The RUBOUT key deletes the last character typed on the current line of keyboard input (except in Host mode). A cross-hatched rectangle, , appears over the deleted character and the alphanumeric cursor moves to the same position in the next line when the next character is typed.

Example

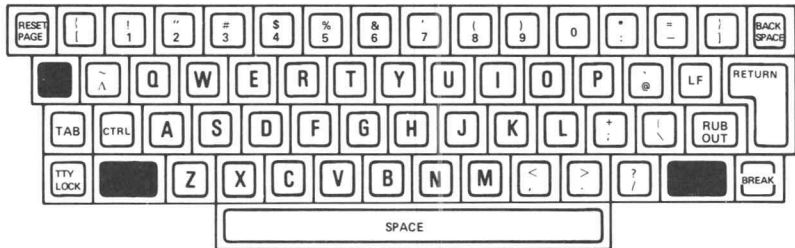
```
#Say help
lo to 
your computer
```

SHIFT

The SHIFT key, when pressed alone, restores the Display Monitor to the View state from the Hold state (identical to pressing the VIEW button on the front panel of the Storage Display Monitor). When a character key is pressed while depressing the SHIFT key, the character is shifted to upper case.

SHIFT-BREAK

Pressing the BREAK key while depressing the SHIFT key places the 4081 Graphic System in Host mode. The message <Host> is displayed on the screen, the 4081 emulates a computer terminal, and all keyboard input is transmitted directly to the host computer or external device to which the primary communications interface is connected. The primary communications interface must first be properly connected and initialized (see the section **Data Communications In Host Mode**).

SHIFT-ESC

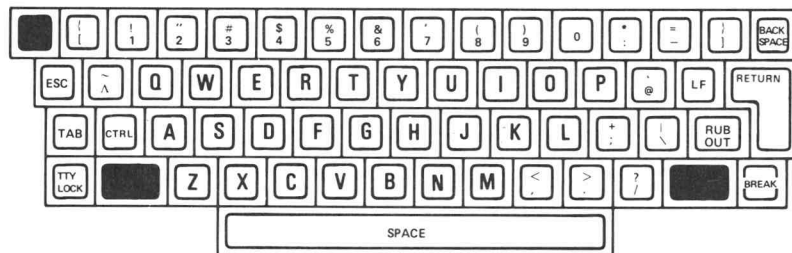
SWITCHES, MESSAGES, MODES, LIGHTS, AND KEYS

Pressing the ESC key while depressing the SHIFT key suspends execution of the program running on the 4081 and places the 4081 Graphic System in Command mode. The message **<Attn>** and the system prompt character **#** are displayed on the screen.

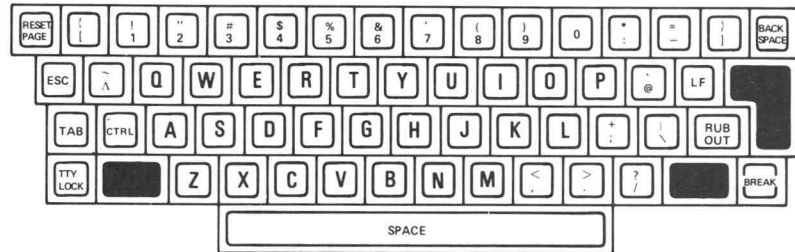
Example

The operator wishes to change the Display Monitor character size to size 3 while a program is running on the 4081. Press SHIFT-ESC to suspend the program. Type "CHA 3" to change the character size. Press RETURN. Type "CON" to continue the program from the point of interruption. Press RETURN. Any characters that the program displays on the screen are now displayed in character size 3.

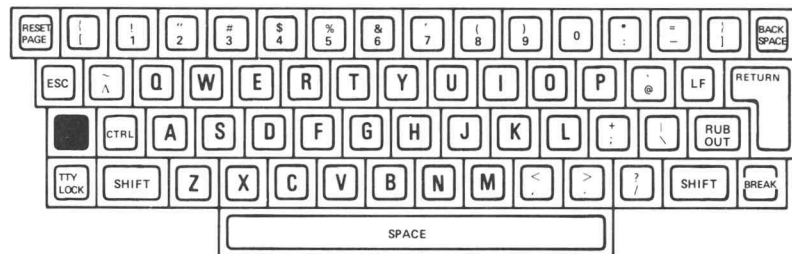
SHIFT-RESET



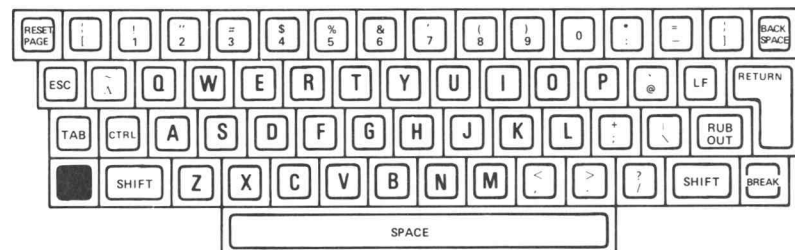
The 4081 Graphic System can be modified, at a customer's request, so that pressing the RESET/PAGE key while depressing the SHIFT key initializes the 4081. Consult a Tektronix Field Service Specialist for further information.

SHIFT-RETURN

Pressing the RETURN key while depressing the SHIFT key returns the 4081 Graphic System from Host mode to local control and displays the message **<Local>** on the screen. If a program is running on the 4081, control is returned to the program (the 4081 enters Program mode). If no program is running, control is returned to the GOS command processor (the 4081 enters Command mode).

TAB

The TAB key has no function except when assigned a special function by a program running on the 4081 (for example, the Plot 80: GOS TECO program).

TTY LOCK

Pressing the TTY LOCK key causes it to lock in the down position. In this position, only the 64 TTY printing character set of the ASCII code can be typed. The 64 TTY printing character set contains the following characters:

SWITCHES, MESSAGES, MODES, LIGHTS, AND KEYS

SP | " # \$ % & ' () * + , - . /
Ø 1 2 3 4 5 6 7 8 9 : ; < = > ?
@ A B C D E F G H I J K L M N O
P Q R S T U V W X Y Z [\] ^ _

Pressing the TTY LOCK key again releases it so that the full 96 printing character set of the ASCII code can be typed. (See the appendix **ASCII Code Chart**.)

SECTION 8

FILES

CONTENTS

	Page
File-Structured Devices	8-3
The Cartridge Tape Unit	8-3
Characteristics	8-3
Inserting a Tape Cartridge	8-6
Write-Protecting a Tape Cartridge	8-7
The Flexible Disc Unit	8-8
Characteristics	8-8
Inserting a Flexible Disc	8-11
Write-Protecting a Flexible Disc	8-13
Copying the System Utilities Tape to a Flexible Disc	8-17
The Hard Disc Unit	8-18
Characteristics	8-18
Inserting a Hard Disc Cartridge	8-20
Write-Protecting a Hard Disc	8-27
Removing a Hard Disc Cartridge	8-28
Copying the System Utilities Tape to a Hard Disc	8-29
A Quick Comparison of File-Structured Mediums	8-30
File Specifiers	8-30
The Device	8-31
The Filename	8-32
The Extension	8-32
Specifying Files Within Library Files	8-35
Device Specifiers	8-37
Formatting a Mass Storage Medium	8-39
Formatting Procedure	8-39
Structure of the Formatted Medium	8-42
Formatting a Library File	8-46
Formatting Procedure	8-46
Structure of the Library File	8-50
Formatting Library Files Within Library Files	8-53
File Processing	8-58
The Logical Unit	8-58
Creating a File	8-59
Processing a Permanent File	8-60
File Sizing	8-61

Section 8

FILES

A file is a named collection of data stored on a mass storage medium (a tape cartridge, flexible disc, or hard disc). The data consists of text, picture, or program information generated using the 4081 Graphic System.

FILE-STRUCTURED DEVICES

A file-structured device is a mass storage device connected to the 4081. The device is designed to allow a 4081 operator to first format a mass storage medium and then to create and access files on the formatted medium.

A formatted mass storage medium is called a file structure because it has the capacity to contain a number of files that are accessible by name. The device in which the formatted medium is inserted is called a file-structured device. GOS recognizes only a standard file structure. A standard file structure is created by formatting a medium using the FORMAT utility program. Only devices that are capable of containing the GOS standard file structure are considered file-structured devices.

Three file-structure devices are available with the 4081: the Cartridge Tape Unit, the Flexible Disc Unit, and the Hard Disc Unit. One Cartridge Tape Unit is part of the standard 4081 Graphic System. Both the Flexible Disc and Hard Disc Units are available alone or in combination as options for the 4905 Mass Storage Module.

THE CARTRIDGE TAPE UNIT

Characteristics

The Cartridge Tape Unit is located in the left cabinet of the 4081 (see Fig. 8-1). There are a maximum of two Cartridge Tape Unit drives available. The first drive (in the top half of the cabinet) is a standard feature of the 4081. It contains circuitry that aids in loading the data on the GOS IPL tape cartridge into memory. (The GOS IPL tape cartridge contains the Graphic Operating System program.) An optional second Cartridge Tape Unit drive can be added to the 4081 (in the bottom half of the cabinet).

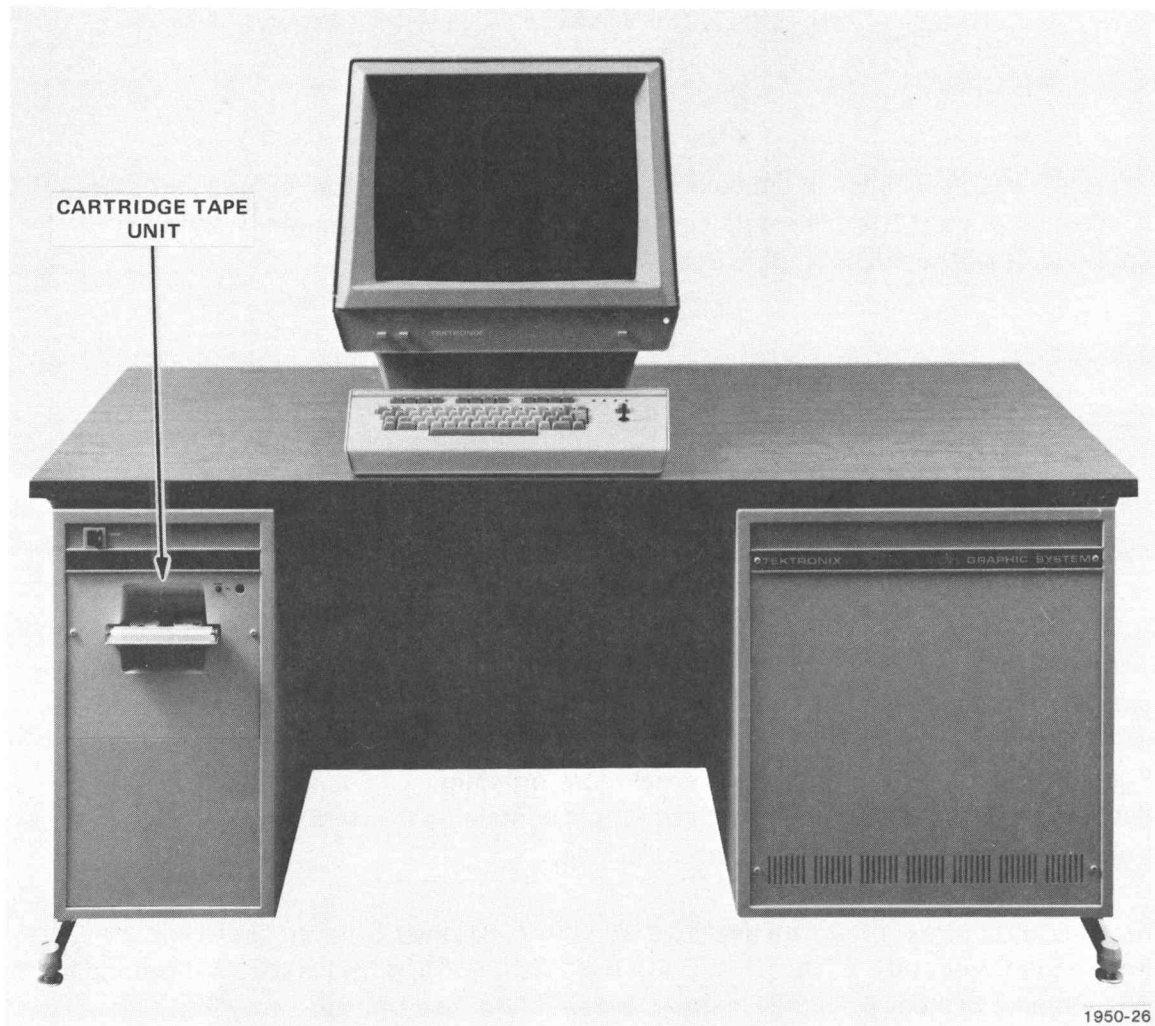


Fig. 8-1. The Cartridge Tape Unit.

The Cartridge Tape Unit uses the Tektronix Tape Cartridge (Fig. 8-2). The cartridge is a magnetic tape storage medium designed for digital applications. The base of the cartridge is heavy-gauge metal; the remainder of the cartridge is a high-impact plastic. Each tape cartridge provides 300 feet of usable storage space.

Formatting a tape cartridge using the FORMAT utility program takes approximately five minutes. A formatted tape cartridge contains 1100 blocks, including the blocks reserved for the primary directory. Each block has a capacity of 256 bytes of data storage.

The data transfer rate indicates how quickly data is read from and written to a device connected to the 4081. The data transfer rate for the Cartridge Tape Unit is measured in terms of the number of data bits (binary digits) that are transferred between a tape cartridge and the 4081 processor in one second. The maximum data transfer rate for the

Cartridge Tape Unit is approximately 38 kilobits/second (38,000 bits per second). In one second, 38,000 bits can be read from a tape cartridge by the 4081 processor; or, in one second, 38,000 bits can be written to a tape cartridge by the 4081 processor.

Because of mechanical limitations, processing files on a tape cartridge takes more time than processing files on a flexible disc or a hard disc. The limitations are obvious in a situation where the tape in a cartridge is near the end of the reel and the operator wishes to create a new file on the cartridge. The tape must first rewind to the primary directory at the beginning of the tape (which take 40 seconds). At this point, a directory entry is created for the new file. Next, the tape must advance to an empty space on which to store the new file. The empty space could be located near the end of the tape, close to the position of the tape before the new file was created. After the new file is closed, the tape again rewinds to the beginning so that the final size of the new file can be added to its directory entry. The entire process requires several minutes and three passes of almost the entire length of the tape. On a disc, the same process might require only several seconds.



Fig. 8-2. The Tektronix Tape Cartridge.

Inserting a Tape Cartridge

1. Pick up the tape cartridge with your fingers supporting the metal base of the cartridge, and your thumb pressing against the clear plastic side of the cartridge.
2. Insert the tape cartridge into the slot in the Cartridge Tape Unit drive. The label on the cartridge should be facing away from the 4081 (Fig. 8-3).

When the tape cartridge is properly inserted, the Cartridge Tape Unit automatically rewinds the cartridge to the beginning of the tape. The tape takes a maximum time of 40 seconds to rewind. While the tape is rewinding, the red BUSY light is lit. The BUSY light is located on the upper right of each Cartridge Tape Unit drive (Fig. 8-4). The BUSY light is lit whenever the tape in the cartridge is moving.



Fig. 8-3. Inserting a tape cartridge into the Cartridge Tape Unit drive.

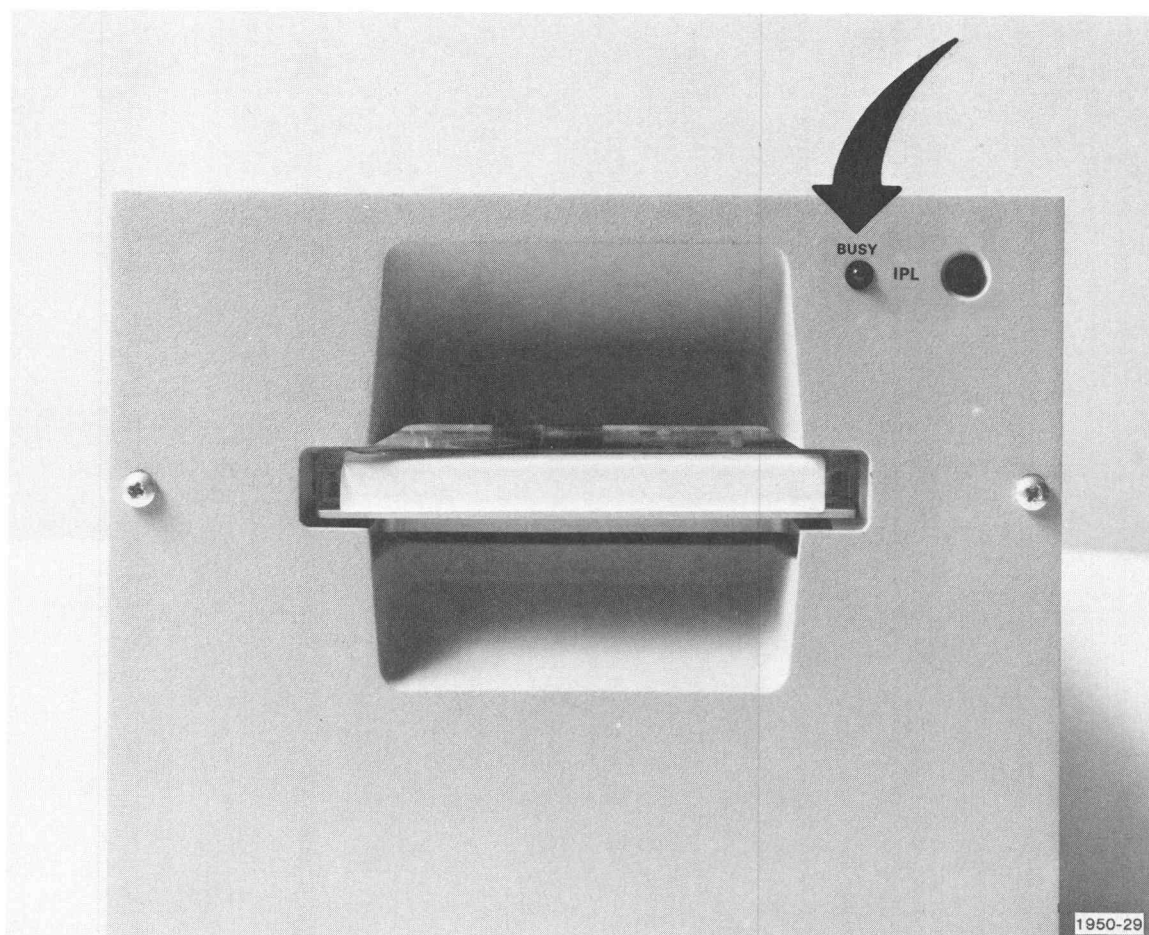


Fig. 8-4. The BUSY light on the Cartridge Tape Unit drive.

Write-Protecting a Tape Cartridge

It is possible to prevent the writing of data on the tape. This is called write-protecting a cartridge. To write protect a cartridge, insert a screwdriver or coin into the write-protect cylinder. Turn the cylinder until the arrowhead points to the position marked SAFE (Fig. 8-5). The cylinder will lock into this position. To remove the write-protection from a cartridge, turn the write-protect cylinder until the arrowhead points to the position opposite SAFE. The cylinder also will lock into this position.



Fig. 8-5. Write-protecting the tape cartridge.

THE FLEXIBLE DISC UNIT

Characteristics

The Flexible Disc Unit is available as an option for the 4905 Mass Storage Module. The Flexible Disc Unit can be ordered with either two or four drives (Fig. 8-6). Each drive has a capacity of one flexible disc at a time.

The Flexible Disc Unit uses the Tektronix Flexible Disc (Fig. 8-7). It is a paper-thin disc coated with a magnetic oxide. In appearance, the flexible disc itself resembles a small 45 rpm phonograph record. The disc is enclosed in a non-removable plastic jacket. The jacket protects the disc during handling, operation and storage.

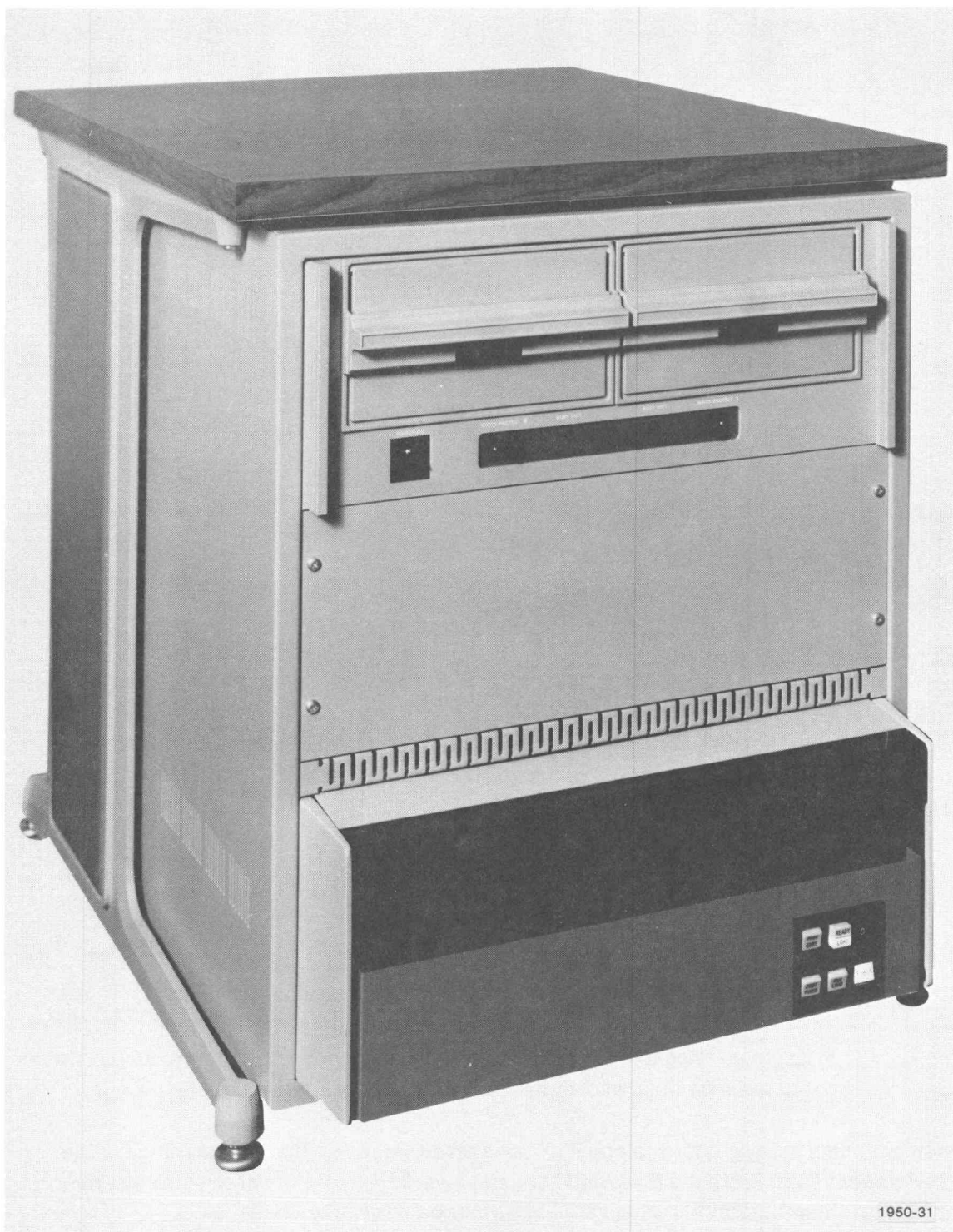


Fig. 8-6. The 4905 Mass Storage Module option combination 31, 33 (with two Flexible Disc Unit drives).

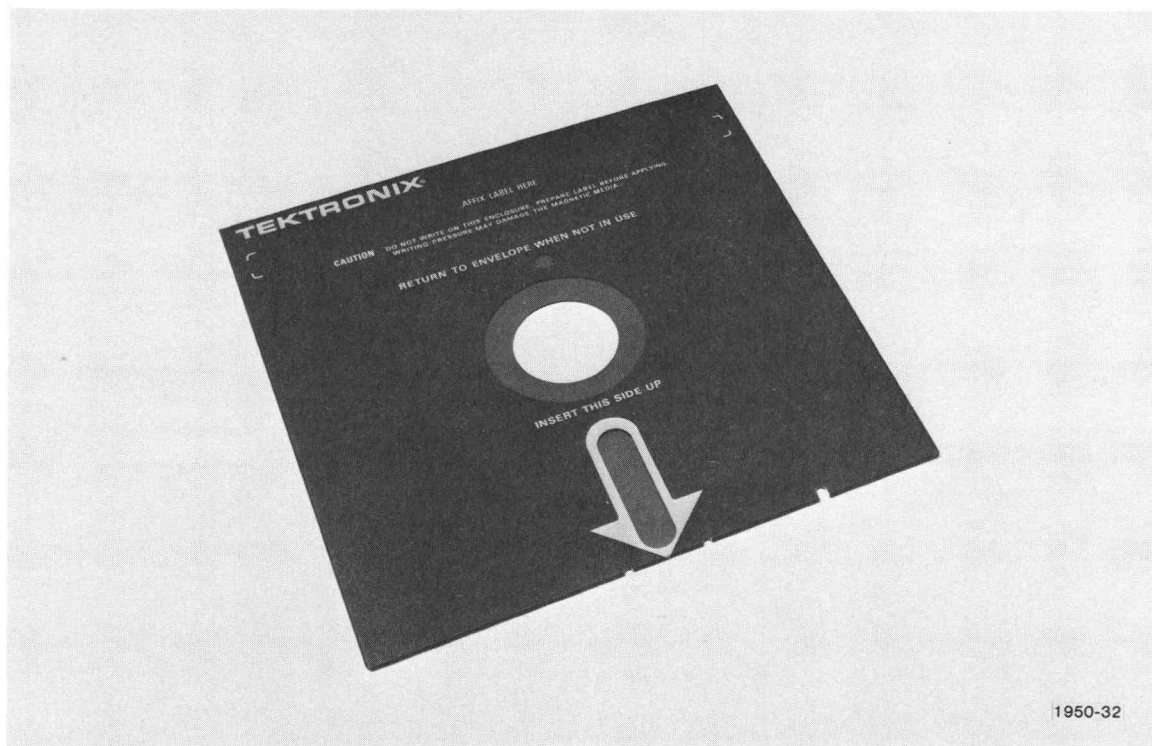


Fig. 8-7. The Tektronix Flexible Disc.

Despite the protection provided by the jacket, a flexible disc should be treated with care. One small fingerprint on the surface of the disc could destroy several files. When handling a flexible disc, try not to:

- Touch the disc through the openings in the jacket.
- Place heavy objects on the disc.
- Write on the disc (write on the label before applying it to a disc).
- Expose a disc to heat, strong sunlight or magnetic fields (for example, do not put a flexible disc on the surface of the Graphics Tablet).

When not using a flexible disc, place it in its paper envelope so that the openings in the jacket are covered. Store the disc in the envelope vertically to prevent dust from settling on the disc. Do not attempt to clean a disc; further damage may result.

Formatting a flexible disc using the FORMAT utility program takes approximately one minute and ten seconds. A formatted flexible disc contains 1232 blocks, including the blocks reserved for the primary directory. Each block has a capacity of 256 bytes of data storage.

The data transfer rate indicates how quickly data is read from and written to a device connected to the 4081. The data transfer rate for the Flexible Disc Unit is measured in terms of the number of data bits (binary digits) that are transferred between a flexible disc and the 4081 memory in one second. The maximum data transfer rate for the Flexible Disc Unit is approximately 250 kilobits/second (250,000 bits per second). In one second, 250,000 bits can be read from a flexible disc and stored in the 4081 memory; or, in one second, 250,000 bits stored in the 4081 memory can be written to a flexible disc.

Processing files on a flexible disc takes less time than processing files on a tape cartridge. Notice that the data transfer rate alone for the Flexible Disc Unit is more than six times greater than the data transfer rate for the Cartridge Tape Unit. In addition, even before any transfer of data occurs, the file that contains or will contain the transferred data must be accessed. It takes much less time to access a file on a flexible disc than a file on a tape cartridge.

The mechanical action of a cartridge-type tape is one reason for the difference in access times between the two devices. To access a file on a tape cartridge, it is necessary to move the tape until the file is positioned in front of the tape head. However, the read/write head on the Flexible Disc Unit moves. In addition, the flexible disc rotates in the drive. Therefore, to access a file on a flexible disc, both the read/write head and the disc move so that the file is positioned in front of the read/write head much more quickly.

Inserting a Flexible Disc

1. If the Flexible Disc Unit is off, press the top half of the POWER ON switch. There is one POWER ON switch for each drawer of two Flexible Disc Unit drives. The green light in the switch should be lit (Fig. 8-8). The MAIN POWER switch on the lower rear side of the 4905 Mass Storage Module should be in the ON position and the green light in the switch lit.
2. If the door on the Flexible Disc Unit drive is closed, open it by pressing up on the black latch and raising the door (Fig. 8-9).
3. Pick up the flexible disc so that the label side of the jacket faces up. On Tektronix discs, the instruction INSERT THIS SIDE UP is printed on the disc jacket. Slide the disc into the slot in the drive. Insert the edge with the write-protect notch first. (The write-protect notch is often covered with a small strip of tape.) On Tektronix discs, an arrow printed on the disc jacket points in the direction that the disc is inserted (Fig. 8-10).
4. Close the door by lowering it until it locks into place.

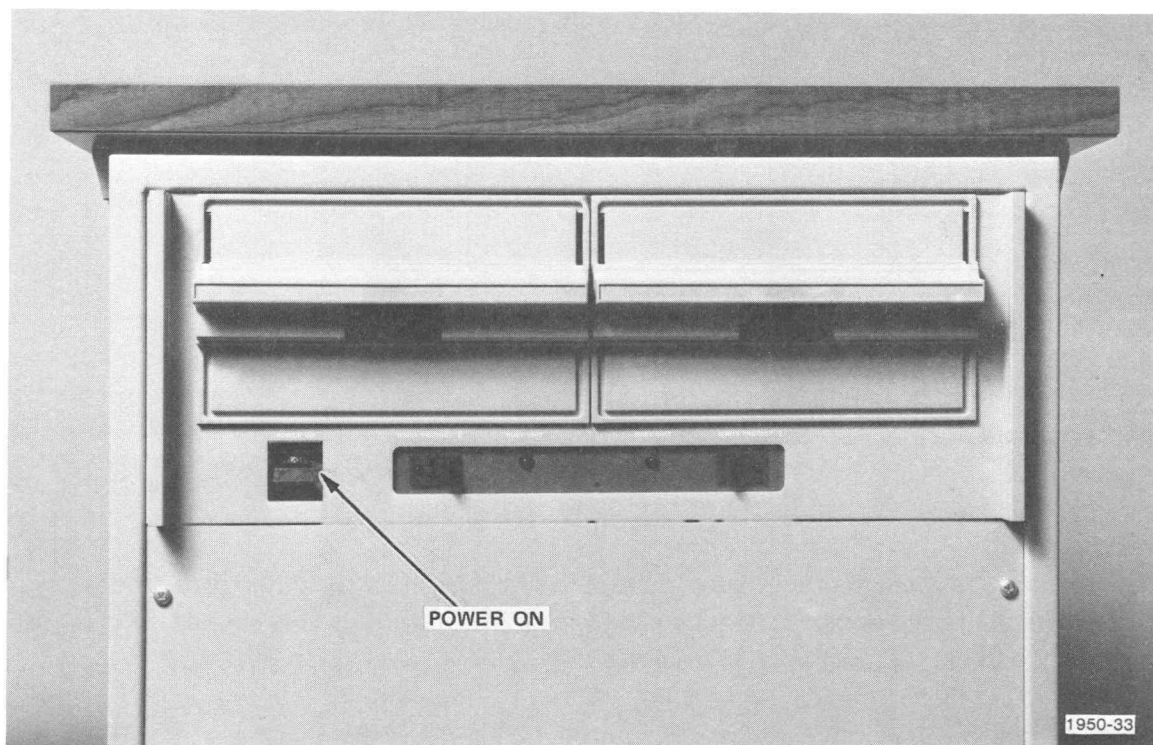


Fig. 8-8. The green light in the POWER ON switch is lit when the Flexible Disc Unit is on.

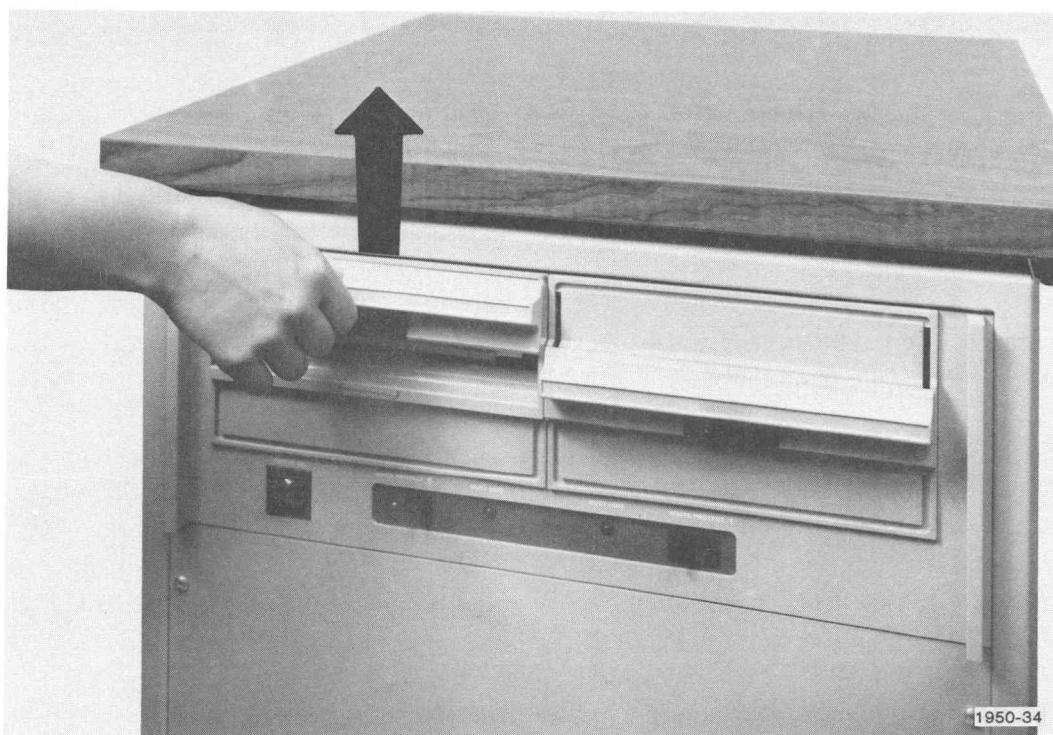


Fig. 8-9. Opening the door on the Flexible Disc Unit drive.

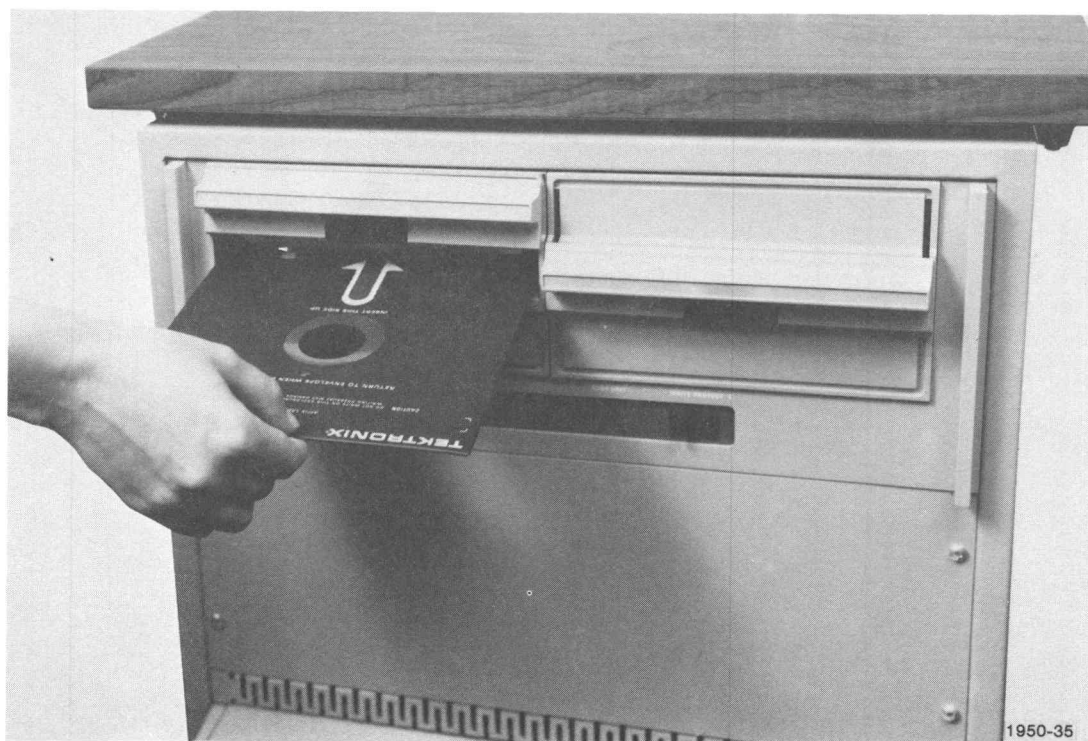


Fig. 8-10. Inserting a flexible disc into the Flexible Disc Unit drive.

The flexible disc starts rotating in the drive as soon as the door is closed. Before accessing information on the disc from the 4081, allow a few seconds for the disc to reach its proper rotational speed.

When accessing information on the disc, notice that the red BUSY UNIT light on the drive is lit. Don't worry if it sounds like a cricket is trapped in the Flexible Disc Unit drive; this is the normal clicking sound of the read/write head movement.

Write-Protecting a Flexible Disc

There are two ways to prevent the writing of data on a flexible disc. One way is to write-protect the Flexible Disc Unit drive in which the disc is inserted. The other is to write-protect the flexible disc itself.

There is a small u-shaped notch in the edge of each flexible disc. On Tektronix discs, the notch is located in the corner of the jacket that is diagonally opposite the TEKTRONIX trademark (Fig. 8-11). If the notch is left uncovered when the disc is inserted in the drive, no data can be written on the disc; the disc is write-protected.

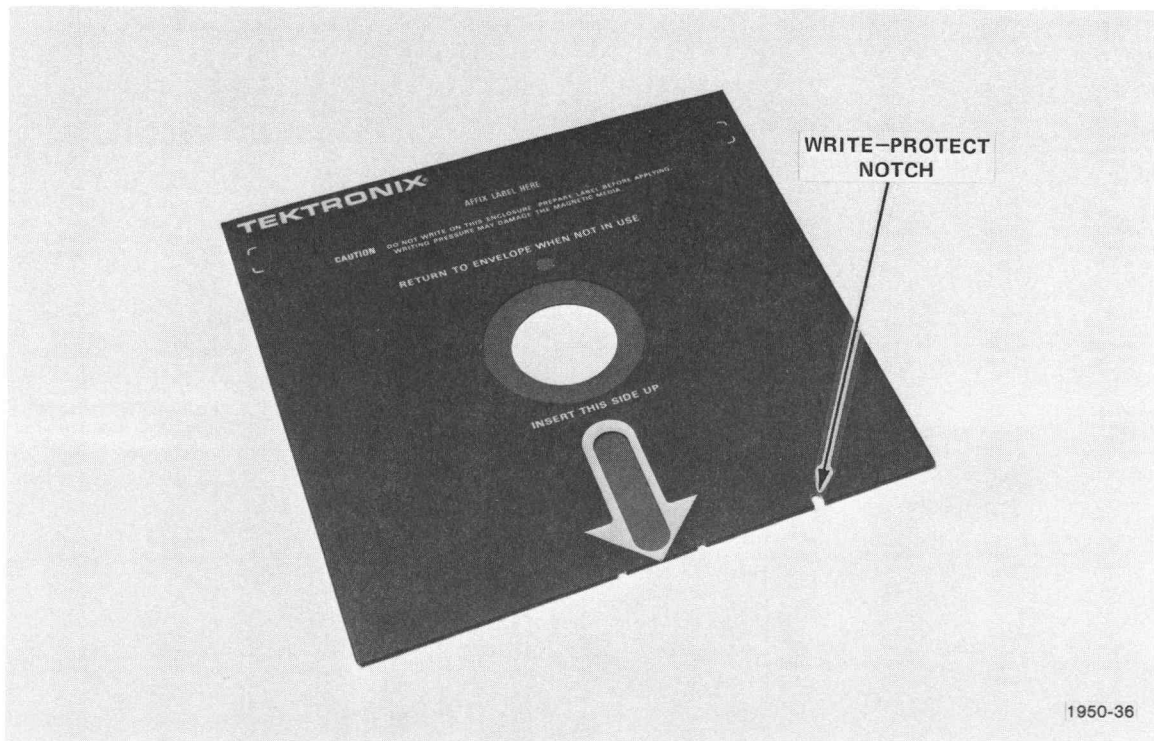


Fig. 8-11. The write-protect notch in the flexible disc.

The red WRITE PROTECT light on the drive remains lit when the disc is inserted and the door is closed. (The red WRITE PROTECT light is also lit when there is no flexible disc inserted in the drive.)

To allow data to be written on the disc, the write-protect notch must be covered with a material dark enough or thick enough to block the passage of light through the notch.

A box of Tektronix flexible discs includes several small strips of adhesive tape. To remove the write-protection from a flexible disc, peel the tape from its backing and apply it to the flexible disc jacket so that both sides of the write-protect notch are covered (Fig. 8-12). The tape can be easily removed if the operator wants to write-protect the disc again.

The operator can keep the write-protect notch covered with tape and still prevent the writing of data on the disc. This is done by write-protecting the Flexible Disc Unit drive in which the disc is inserted. There is a black switch marked WRITE PROTECT on each drive. To write-protect the drive, flip the switch toward the red WRITE PROTECT light. The light should be lit (Fig. 8-13).

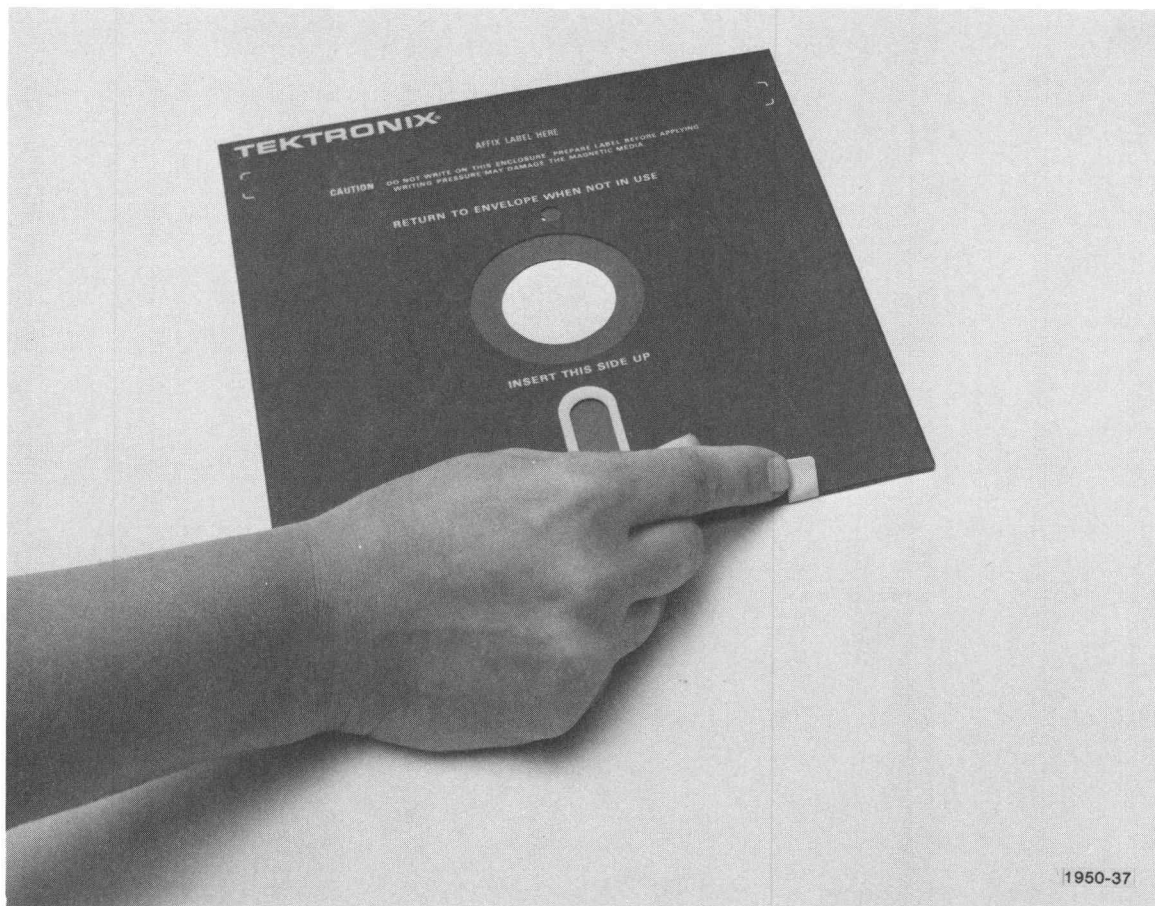


Fig. 8-12. Covering the write-protect notch with tape.

Of the two ways to write-protect a flexible disc, uncovering the write-protect notch in the disc has priority. If the write-protect notch in a flexible disc is uncovered, the disc is write-protected whether the WRITE PROTECT switch on the drive is on or off. If the write-protect notch in the disc is covered, however, the operator can more conveniently write-protect a disc or remove the write-protection from a disc by turning the WRITE PROTECT switch on or off.

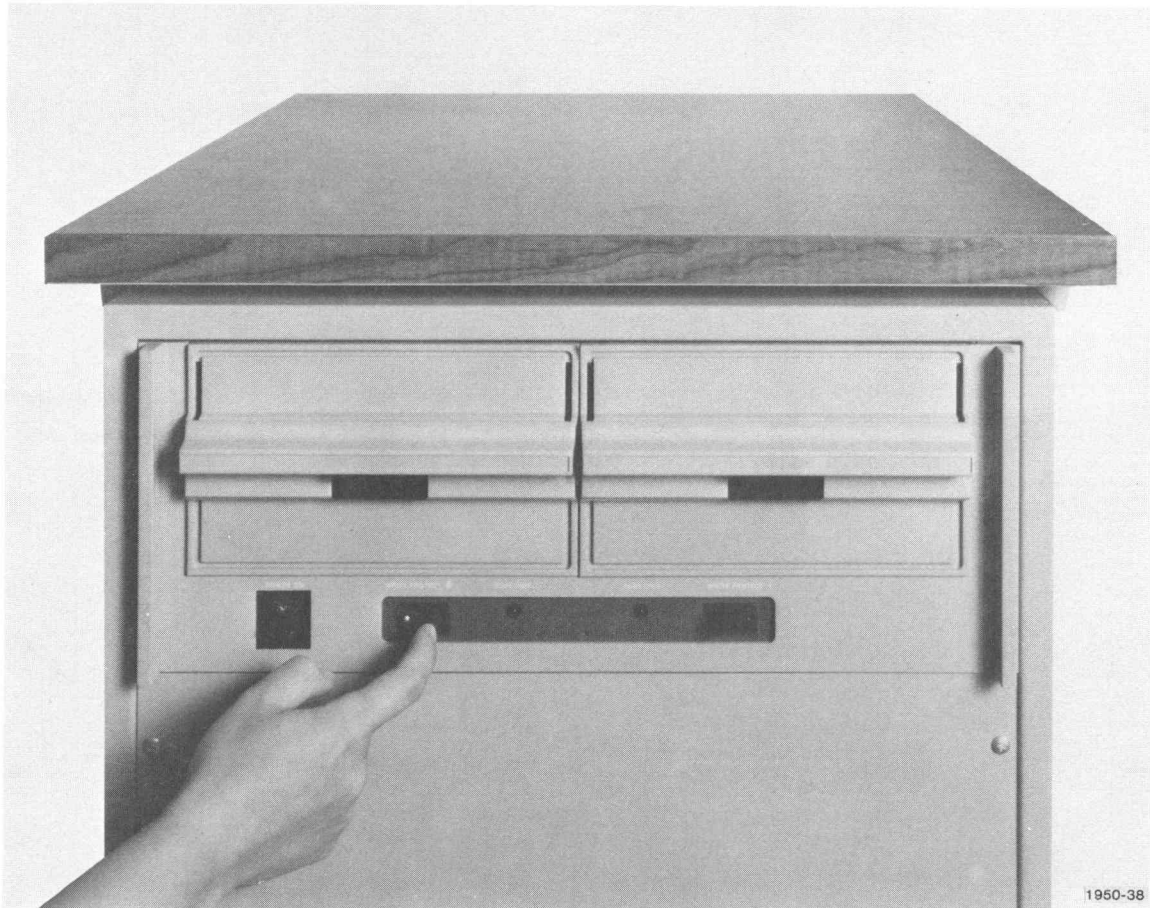


Fig. 8-13. Write-protecting the Flexible Disc Unit drive.

For example, a programmer has a collection of system program files on the flexible disc inserted in the second Flexible Disc Unit drive. The write-protect notch in the disc is covered with tape. The programmer then uses a scratch disc inserted in another drive to create and debug a new system program file. While creating the new file, the programmer flips the WRITE PROTECT switch on so that none of the system program files are accidentally destroyed. After the new system program file is running smoothly, the programmer flips the WRITE PROTECT switch off. Then, using the COPY utility program, the new system program file can be added to the collection of system program files on the disc in the second Flexible Disc Unit drive.

NOTE

Remember that the only situation in which data can be written on a flexible disc is when the write-protect notch in the disc is covered and the drive's WRITE PROTECT switch is off.

Copying the System Utilities Tape to a Flexible Disc

The GOS utility programs that come with the standard 4081 Graphic System are stored in files on the system utilities tape cartridge. Because of the added speed and convenience of the Flexible Disc Unit, it is preferable to copy the files on the system utilities tape to a flexible disc. The following procedure describes how to copy the files:

1. Ensure that both the 4081 and the 4905 Mass Storage Module are turned on.
2. Load the GOS program into the 4081 memory by inserting the GOS IPL tape cartridge in the top Cartridge Tape Unit drive and pressing the IPL button.
3. After the GOS program is loaded into memory, remove the GOS IPL tape cartridge from the top Cartridge Tape Unit drive and insert the system utilities tape cartridge.
4. Insert a flexible disc in the first Flexible Disc Unit drive (the left-most drive on the 4905 Mass Storage Module). Make sure that the write-protect notch in the disc is covered with tape and that the drive's WRITE-PROTECT switch is off. Also make sure that the drive door is closed.

5. Type,

```
#CT0:FORMAT FD0:;N:SYSTEM
```

then press the RETURN key. The following message is displayed:

Mount volume to be formatted

Type a "G" to Go

6. After the message **Formatting Done** is displayed, type,

```
#CT0:COPY FD0:=CT0:
```

then press the RETURN KEY. The files on the system utilities tape cartridge are displayed on the screen one-by-one as each is copied to the flexible disc. When the # prompt character is displayed on the screen, all the files on the system utilities tape cartridge have been copied to the flexible disc. The flexible disc in the first Flexible Disc Unit drive is now the system disc (because it contains the GOS utility programs).

THE HARD DISC UNIT

Characteristics

The Hard Disc Unit is available as an option for the 4905 Mass Storage Module. The Hard Disc Unit can be ordered with either one or two drives (see Fig. 8-14). There are two hard discs for each drive: a non-removable disc (the bottom disc) and a disc cartridge (the top disc).

The Hard Disc Unit uses the Tektronix Hard Disc Cartridge (Fig. 8-15). It is an aluminum disc coated with a magnetic oxide. In appearance, the hard disc itself resembles a long-playing record album. The disc is enclosed in a plastic cover for protection when not in use. The bottom half of the cover is removed when the disc is inserted in the drive.

Formatting a hard disc using the FORMAT utility program takes approximately three minutes. A formatted hard disc contains 19,584 blocks, including the blocks reserved for the primary directory. Each block has a capacity of 256 bytes of data storage.

The data transfer rate indicates how quickly data is read from and written to a device connected to the 4081. The data transfer rate for the Hard Disc Unit is measured in terms of the number of data bits (binary digits) that are transferred between a hard disc and the 4081 memory in one second. The maximum data transfer rate for the Hard Disc Unit is approximately 2500 kilobits/second (2,500,000 bits per second). In one second, 2,500,000 bits can be read from a hard disc and stored in the 4081 memory; or, in one second, 2,500,000 bits stored in the 4081 memory can be written to a hard disc.

Processing files on a hard disc takes less time than processing files on a flexible disc or tape cartridge. The data transfer rate for the Hard Disc Unit is ten times greater than the data transfer rate for the Flexible Disc Unit and more than 65 times greater than the rate for the Cartridge Tape Unit. It also takes much less time to access a file on a hard disc or tape cartridge. On the average, a file on a hard disc is accessed nearly ten times faster than a file on a flexible disc.

Both the Hard Disc Unit and Flexible Disc Unit operate in a similar manner. The disc rotates in the drive and the read/write head moves to access a file. Processing files on the hard disc, however, takes less time than processing files on a flexible disc. An important reason for the difference in file processing time between the two devices is the difference in rotational speeds. A hard disc rotates in the Hard Disc Unit drive nearly seven times faster than a flexible disc rotates in the Flexible Disc Unit drive.



Fig. 8-14. The 4905 Mass Storage Module option combination 31, 33 (with one Hard Disc Unit).

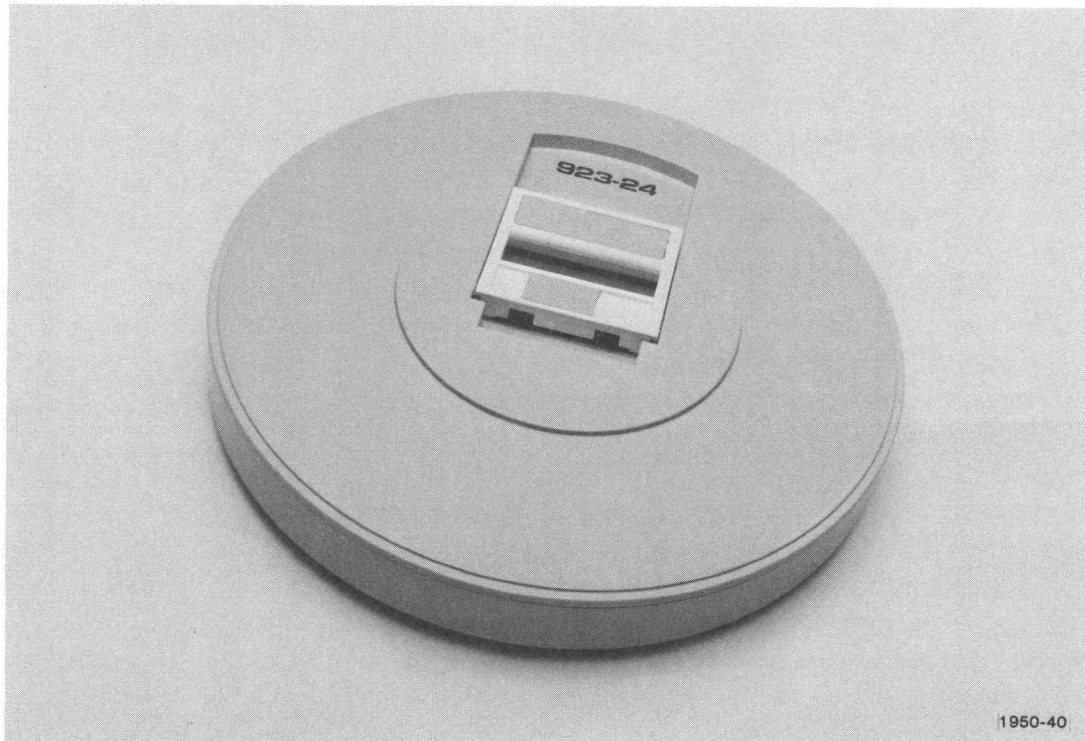


Fig. 8-15. The Tektronix Hard Disc Cartridge.

Inserting a Hard Disc Cartridge

1. If the Hard Disc Unit is off, press the top half of the POWER switch located on the lower right of the drive. There is one POWER switch for each Hard Disc Unit drive. The white light in the switch should be lit (Fig. 8-16). The MAIN POWER switch on the lower rear side of the 4905 Mass Storage Module should be in the ON position and the green light in the switch lit.
2. Press the lower half of the RUN/LOAD switch located to the left of the POWER switch. The switch should be depressed in the LOAD position. Wait approximately 25 seconds until the LOAD half (the lower half) of the white READY/LOAD indicator light is lit (Fig. 8-17).
3. When the LOAD light is lit, place your fingers in the recess beneath the smoke plastic door on the front of the drive. Now pull the drive out from the cabinet as if opening a drawer (Fig. 8-18).

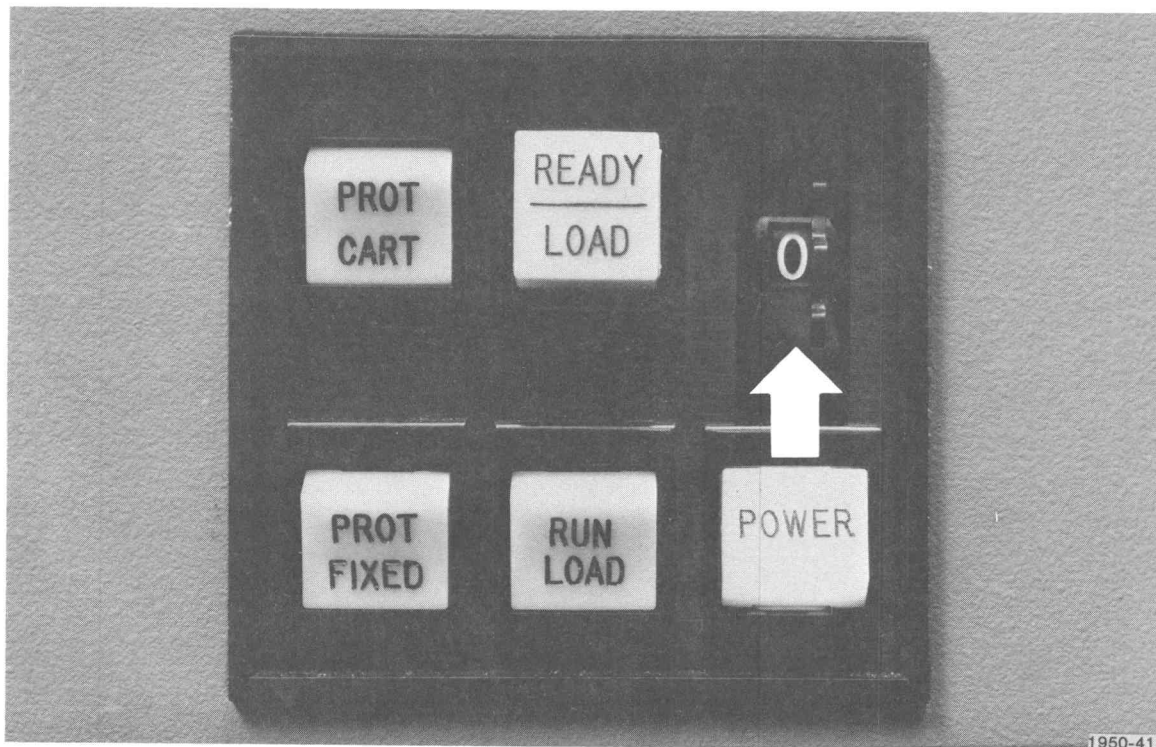


Fig. 8-16. The white light in the POWER switch is lit when the Hard Disc Unit is on.

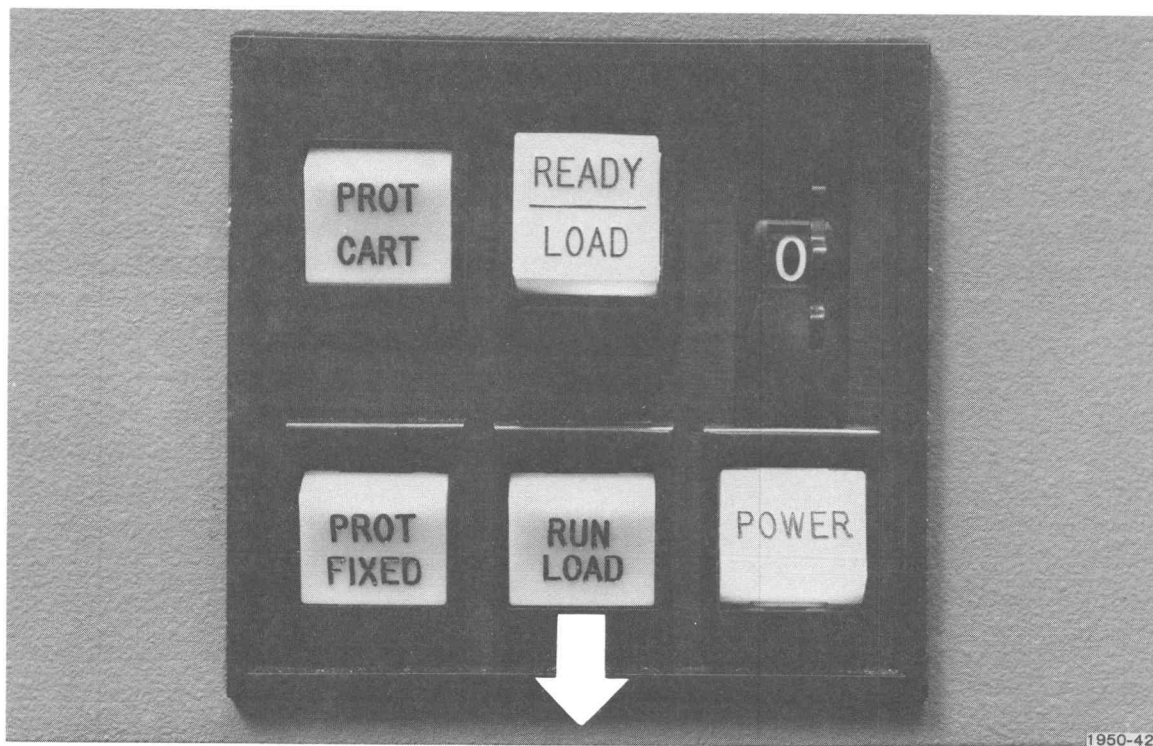


Fig. 8-17. The lower half of the READY/LOAD indicator light must be lit before inserting the hard disc cartridge.

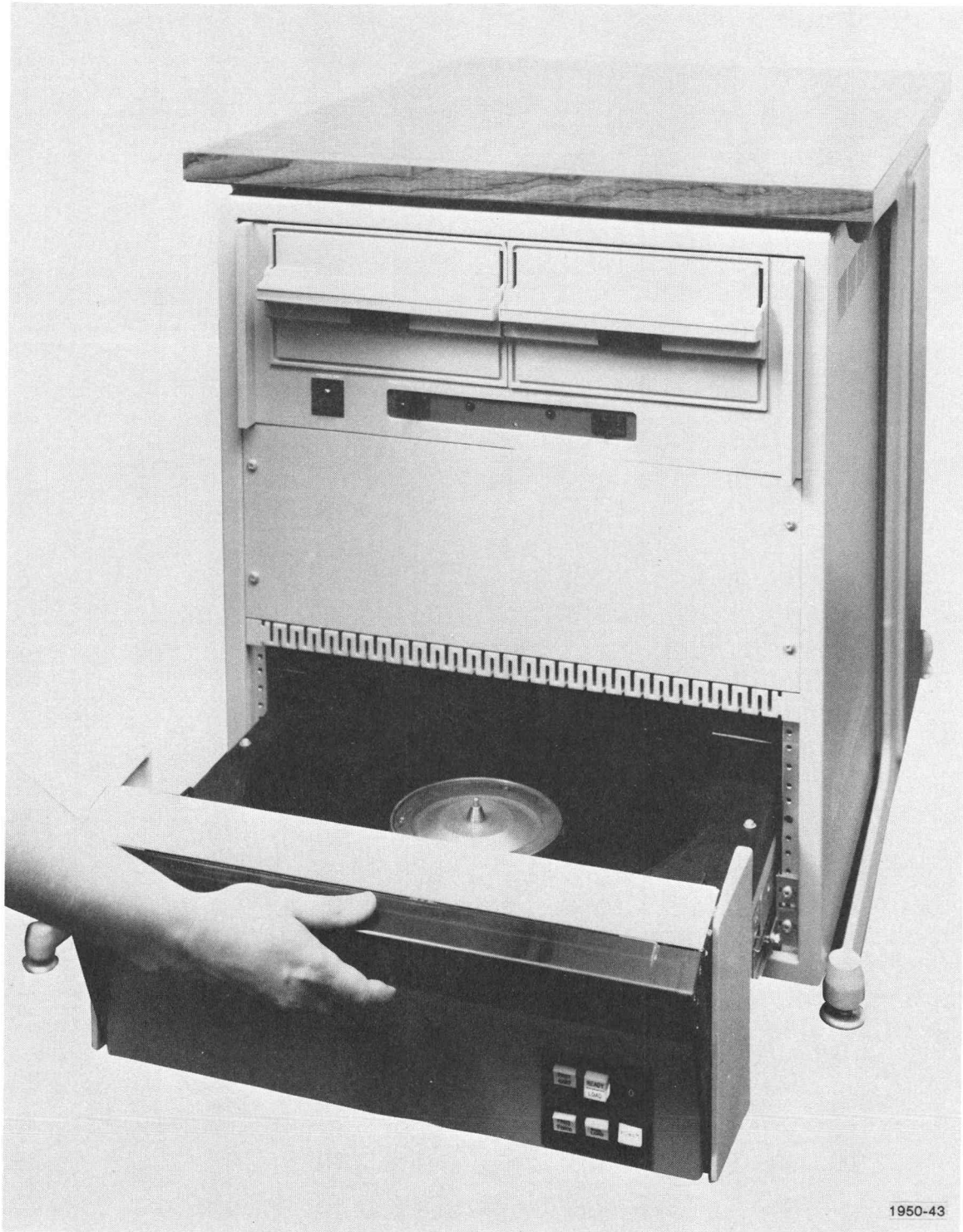


Fig. 8-18. Pulling the Hard Disc Unit drive out from the cabinet.

4. Before inserting the hard disc cartridge, the bottom plastic cover must be removed. While the cartridge handle is down, slide the release button on the handle to the left. While holding the button in this position, lift the handle up (Fig. 8-19). The bottom cover is now released and can be removed. The bottom cover is later placed on top of the loaded cartridge.
5. Place the hard disc cartridge in the drive so that the recess in the top of the cartridge cover is closest to the cabinet (Fig. 8-20). The cartridge can be correctly inserted only in this position. If the cartridge is not correctly inserted, turn it back and forth until it drops into place.
6. Fold the cartridge handle down. Position the bottom cover so that its hollow side faces down, and place it on top of the loaded cartridge (Fig 8-21). The bottom cover acts as a dust cover for the hard disc cartridge.
7. Close the drive door and push the drive back into the cabinet.

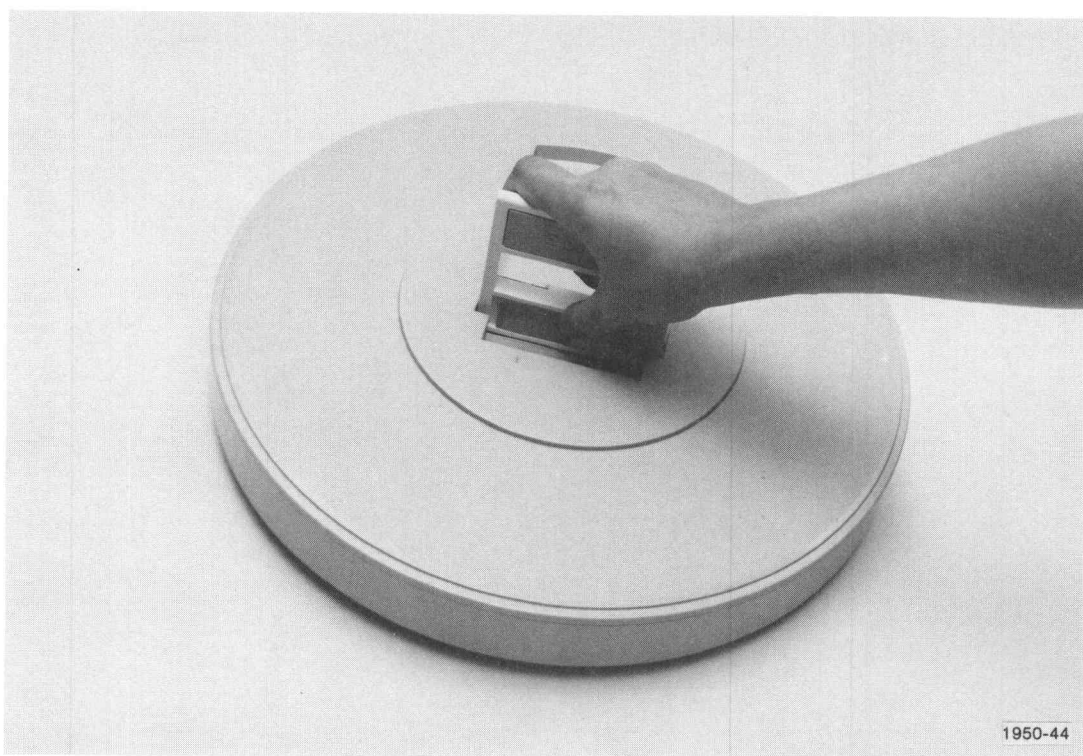


Fig. 8-19. Removing the bottom cover from the hard disc cartridge.



Fig. 8-20. Placing the hard disc cartridge in the Hard Disc Unit drive.



Fig. 8-21. Placing the bottom cover on the inserted hard disc cartridge.

The hard disc cartridge is now inserted in the Hard Disc Unit drive, but the 4081 cannot yet process any files on either the hard disc cartridge or the non-removable hard disc. The operator must first press the top half of the RUN/LOAD switch located near the POWER switch. The switch should be depressed in the RUN position.

As soon as the RUN switch is pressed, the drive door locks and the LOAD half of the READY/LOAD indicator light turns off. Wait approximately 50 seconds until the READY/LOAD indicator light is lit (Fig. 8-22). The READY light indicates that the discs have reached their proper rotational speed. It is now possible to read data from the discs. Notice that the indicator lights on the drive flicker when the discs are rotating at the proper speed.

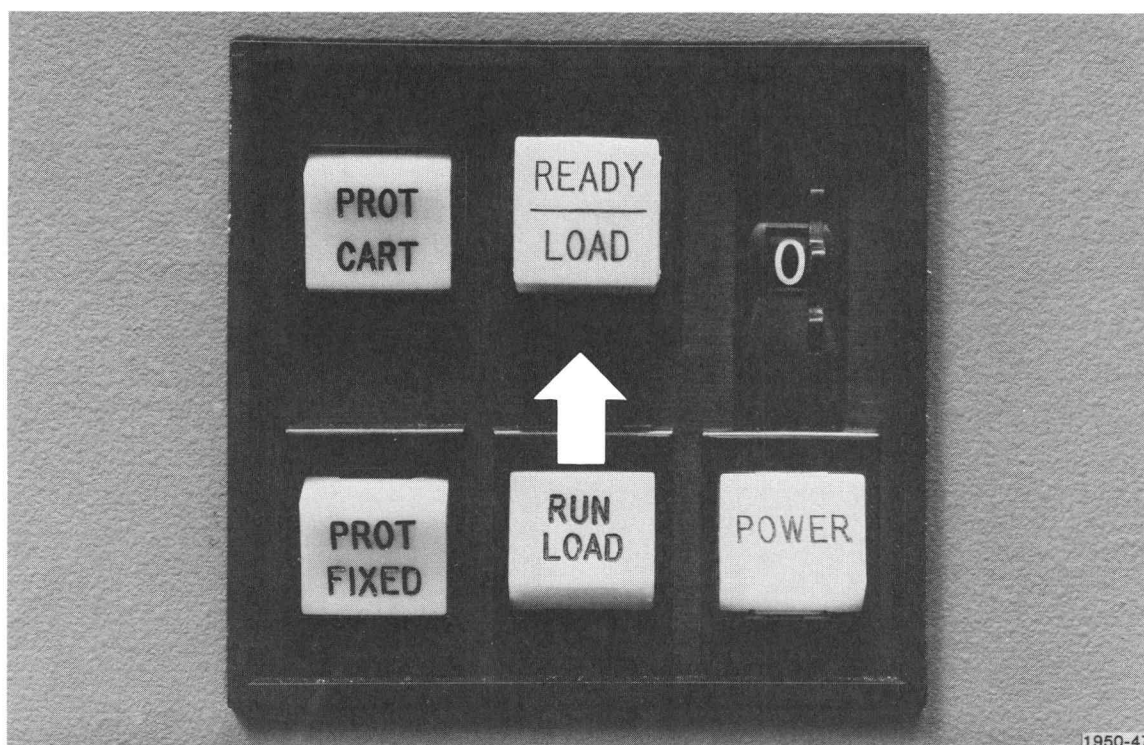


Fig. 8-22. The top half of the READY/LOAD indicator light must be lit before attempting to read data from a hard disc.

To ensure the accuracy of the data written on the hard discs, allow the Hard Disc Unit Drive to reach its proper operating temperature. Wait five minutes after the READY light is lit before creating new files or revising permanent files on the discs in the Hard Disc Unit drive. Also, wait five minutes after the READY light is lit before running the following GOS utility programs if a disc or a file on a disc in the Hard Disc Unit drive is the program argument: ATTRIB, COPY, DELETE, FORMAT, RENAME, and SQUISH.

NOTE

The non-removable disc in a Hard Disc Unit drive cannot be accessed if there is no hard disc cartridge inserted in the drive.

Write-Protecting a Hard Disc

There are two discs in each Hard Disc Unit drive. Each disc is write-protected individually, independent of the other disc. To write-protect the non-removable hard disc (the bottom disc in the drive), press the top half of the PROT FIXED switch located to the left of the RUN/LOAD switch. The white light in the switch should be lit (Fig. 8-23). The non-removable hard disc is write-protected; however, data can still be written on the hard disc cartridge.

To write-protect the hard disc cartridge (the top disc in the drive), press the top half of the PROT CART switch located to the left of the READY/LOAD indicator light. The white light in the switch should be lit (Fig. 8-24). The hard disc cartridge is write-protected; however, data can still be written on the non-removable hard disc. If the PROT CART and the PROT FIXED switches are both lit, then both discs in the Hard Disc Unit drive are write-protected.

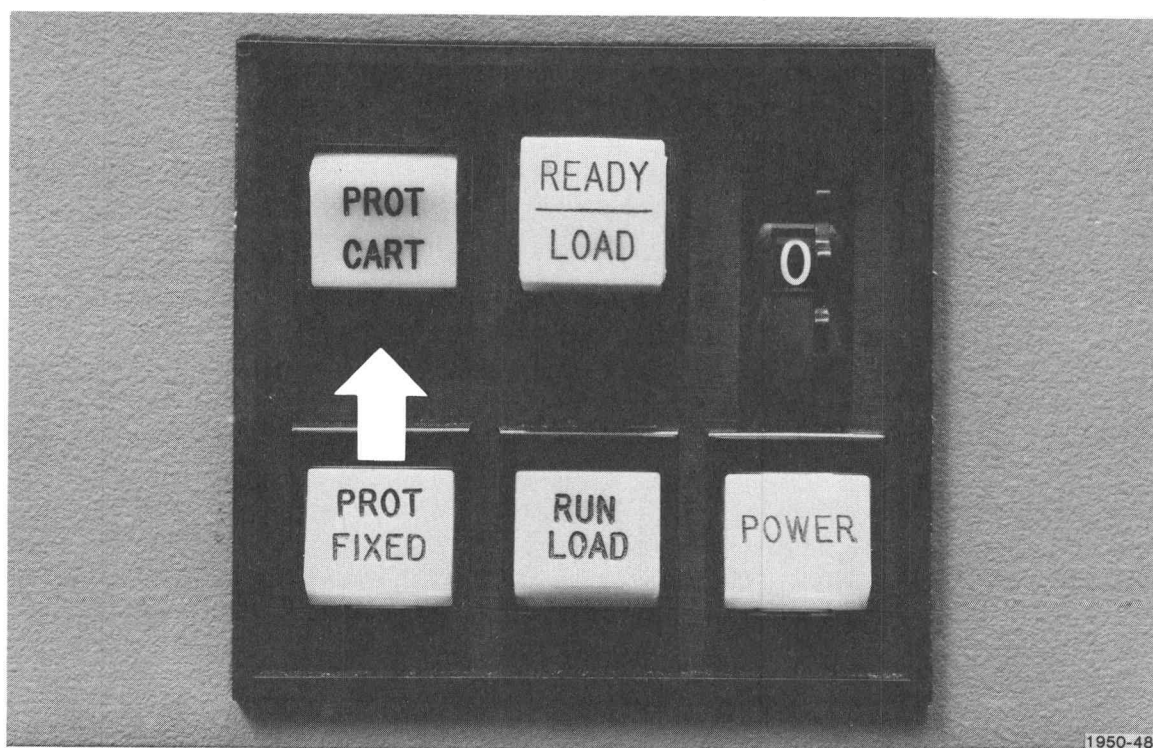


Fig. 8-23. Write-protecting the non-removable hard disc.

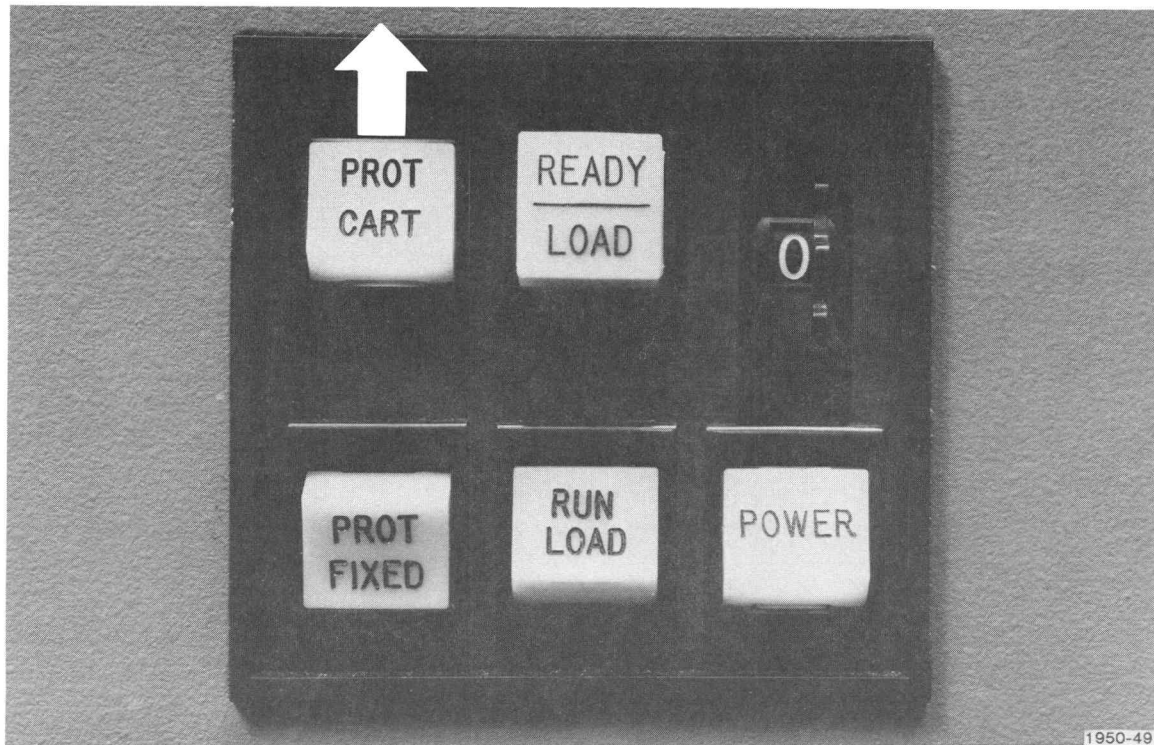


Fig. 8-24. Write-protecting the hard disc cartridge.

Removing a Hard Disc Cartridge

1. Press the lower half of the RUN/LOAD switch. The switch should be depressed in the LOAD position. The READY half of the READY/LOAD indicator light turns off. Wait approximately 25 seconds until the LOAD half of the READY/LOAD light is lit, indicating that the discs have stopped rotating in the drive.
2. Open the drive door and pull the drive out from the cabinet.
3. Remove the bottom cover from the top of the loaded cartridge. While the cartridge handle is down, slide the release button on the handle to the left. While holding the button in this position, lift the handle up. The cartridge now can be removed from the drive. Replace the cartridge in the bottom cover and fold the handle down to lock the bottom cover on to the cartridge.

CAUTION

Do not turn the Hard Disc Unit off unless the LOAD half of the READY/LOAD indicator light is lit. The destruction of data stored on the discs may result.

Copying the System Utilities Tape to a Hard Disc

The GOS utility programs that come with the standard 4081 Graphic System are stored in files on the system utilities tape cartridge. Because of the added speed and convenience of the Flexible Disc Unit, it is preferable to copy the files on the system utilities tape to a flexible disc. The following procedure describes how to copy the files:

1. Ensure that both the 4081 and the 4905 Mass Storage Module are turned on.
2. Load the GOS program into the 4081 memory by inserting the GOS IPL tape cartridge in the top Cartridge Tape Unit drive and pressing the IPL button.
3. After the GOS program is loaded into memory, remove the GOS IPL tape cartridge from the top Cartridge Tape Unit drive and insert the system utilities tape cartridge.
4. Insert a flexible disc in the first Flexible Disc Unit drive (the left-most drive on the 4905 Mass Storage Module). Make sure that the write-protect notch in the disc is covered with tape and that the drive's WRITE-PROTECT switch is off. Also make sure that the drive door is closed.
5. Type,

```
#CT0:FORMAT FD0;;N:SYSTEM
```

then press the RETURN key. The following message is displayed:

Mount volume to be formatted
Type a "G" to Go

6. After the message **Formatting Done** is displayed, type,

```
#CT0:COPY FD0:=CT0:
```

then press the RETURN key. The files on the system utilities tape cartridge are displayed on the screen one-by-one as each is copied to the flexible disc. When the # prompt character is displayed on the screen, all the files on the system utilities tape cartridge have been copied to the flexible disc. The flexible disc in the first Flexible Disc Unit drive is now the system disc (because it contains the GOS utility programs).

A QUICK COMPARISON OF FILE-STRUCTURED MEDIUMS

Medium	Formatting Time	Data Transfer Rate (peak)	Data Capacity (in 256-byte blocks)
Tape Cartridge	5 min.	38 kilobits/sec.	1 100 blocks
Flexible Disc	1 min. 10 sec.	250 kilobits/sec.	1 232 blocks
Hard Disc	3 min.	2500 kilobits/sec.	19,584 blocks

FILE SPECIFIERS

When creating a file on the 4081, the operator assigns a file specifier to the file. The file specifier serves two basic functions: it supplies GOS with the information needed to locate the file for processing and it identifies the contents of the file for the operator.

Under GOS, there is a standard syntax for file specifiers. The standard file specifier consists of three fields: the device, the filename, and the extension. Generally, the operator uses file specifiers when running a program or executing a GOS resident command. The operator can specify a program file to be loaded into memory and run or an argument file to be operated on by a program or resident command. The GOS standard file specifier has the following format:

```
[device:]filename[.extension]
```

THE DEVICE

device: is the 2-3 character mnemonic of the file-structured device in which the mass storage medium containing the file is inserted. For example, for a file that resides on a flexible disc inserted in the second (right) Flexible Disc Unit drive, the device mnemonic is FD1:. The device mnemonic must end with a colon (:).

Table 8-1 lists the GOS device mnemonics for each file-structured device.

Table 8-1
DEVICE MNEMONICS

Device	Mnemonic
Cartridge Tape	CT:(CT0:) CT1:
Flexible Disc	FD:(FD0:) FD1: to FD3:
Hard Disc	DK:(DK0:) DK1: to DK7:
System Device	SYS:
User Device	USR:

In Table 8-1, the first mnemonic listed for each device represents the first device drive. For example, the first Cartridge Tape Unit drive is CT:(or CT0:). The other mnemonics listed for each device represent additional device drives. For example, the third Flexible Disc Unit drive is FD2:.

In the case of the Hard Disc Unit, there are two discs for each drive: a non-removable disc(the bottom disc) and a disc cartridge(the top disc). The non-removable disc of the first Hard Disc Unit drive is DK:(or DK0:). The disc cartridge is DK1:. The non-removable disc of the second Hard Disc Unit drive is DK2:. The disc cartridge is DK3:. There are a maximum of four Hard Disc Unit drives. The disc cartridge of the fourth drive is DK7:.

GOS provides the mnemonics SYS: and USR: as a convenience to the operator. SYS: is generally assigned to the device in which the system disc or tape is inserted. USR: is generally assigned to the device in which the medium containing the user's files is

inserted. Both SYS: andUSR: are automatically assigned to file-structured devices when the GOS program is first loaded into memory. The device assignments are determined by the user when the GOS IPL tape is generated. The operator, however, can reassign the SYS: andUSR: mnemonics to different devices by running the SET utility program.

The device mnemonic field in a file specifier is considered optional. In many cases when the device mnemonic is omitted from a file specifier, a default device is assumed (generally, either the SYS: orUSR: device). The default device is determined by GOS or the program that processes the file.

THE FILENAME

filename is a 1-6 character name assigned to a file by the operator. Only the following characters can be used in a filename (note that a space is not a valid character in a filename):

ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789#\$\$%

A filename aids the operator in identifying the contents of a file. For example, a file containing the text of this manual could be assigned the filename MANUAL. The filename field is required in every file specifier.

The Extension

.extension is a 1-3 character name that is appended to a filename to further identify a file. The extension must be preceded by a period(.). The period separates the extension from the filename.

The extension field in a file specifier is considered optional. In many cases when the extension is omitted from a file specifier, a default extension is assumed. The default extension is determined by GOS or the program that processes the file.

The extension generally identifies the type of file specified. GOS recognizes the following list of standard file extensions. Each standard extension denotes a specific type of file.

.ASM	For assembly language source files. An assembly language source file contains the Plot 80: Assembly Language coding for a program or section of a program. The source program must be converted to a program load file by an assembler before it can be run. .ASM files are stored in ASCII code.
------	---

- .BAD** For files with bad blocks. .BAD files contain one or more bad blocks. A bad block is an area of a mass storage medium found defective by the FORMAT utility program. .BAD files are automatically assigned read and write-protect attributes by the FORMAT program.
- .BAK** For TECO backup files. When a file is edited using the Plot 80: GOS Text Editor & Corrector (TECO), TECO creates, in most cases, a copy(or backup) of the file before it is revised. .BAK files are stored in ASCII code.
- .BAT** For batch files. A batch file contains a sequence of commands to be executed by the GOS command processor. The command sequence can include any combination of GOS resident commands, GOS utility programs, and user programs. Batch files are executed (in other words, each command in the batch file is executed in sequence) by running the BATCH utility program. .BAT files are stored in ASCII code.
- .CMD** For LINK command files. A LINK command file contains a sequence of commands to be executed by the Plot 80: Library Linker/Loader. .CMD files are stored in ASCII code.
- .DAT** For user data files.
- .FOR** For Fortran source files. A Fortran source file contains the Plot 80: Fortran IV coding for a program or section of a program. The source program must be converted to a program load file by a compiler before it can be run. .FOR files are stored in ASCII code.
- .HLP** For help message files. A help message file contains information on the purpose, syntax, and sample uses of a program. Help message files for the GOS utility programs are stored in the library HELP.LIB on the system disc or tape. To display the contents of a .HLP file, run the HELP utility program.
- .LIB** For library files. A library file is a file that has been formatted to contain other files. Library files contain a directory and a collection of the files listed in the directory. To display a list of the files in a .LIB file, run the DIR program.
- .LST** For list files created by the Assembler, the Fortran compiler, and the DIR, HELP, PDBDMP, and SYSTAT utility programs. .LST files are stored in ASCII code.
- .MAC** For macro files. A macro file contains the source code that defines one or more macro instructions. A macro instruction is a programmer-defined Assembler statement that, when processed by the Assembler, generates a sequence of assembly language instructions. A .MAC file is generally copied into a source program file when the program is assembled.

FILES

.MAP	For LINK memory map files. A LINK memory map file contains a list of the program load files (modules) linked into memory by the Plot 80: Library Linker/Loader. .MAP files are stored in ASCII code.
.OBJ	For program load files. A program load file can be loaded into memory and run. The file generally contains a program or section of a program. .OBJ files are created by compiling or assembling a source program file.
.OLY	For LINK overlay files. An overlay file contains a section (overlay) of a program. The program is generally too large to load into memory at once, so, using the Plot 80: Library Linker/Loader, the program is divided into overlays. These overlays are loaded into memory during program execution as they are needed.
.PDB	For Picture Data Base files. A Picture Data Base file contains binary graphics information stored in Plot 80: Picture Data Base format. The graphics information in a .PDB file must be converted to screen coordinates before it can be displayed on a graphic output device.
.PLT	For graphics files. A graphics file contains graphics information in the form of screen X,Y coordinates. .PLT files can be displayed on the screen by running the DISPLA utility program or drawn on a plotter by running the PLOT utility program.
.SAV	For memory image files. A memory image file contains binary code that is copied directly into memory at an absolute location.
.SYS	For system files. Generally, .SYS files are referenced by GOS program files. For example, the GOS Assembler program file ASM.OBJ, when loaded into memory and run, references the files SVC10.SYS and ASMBLR.SYS on the system disc or tape.
.WSP	For Graphic Function Manager (GFM) workspace files. A GFM workspace file contains a list of user-defined values for GFM parameters (for example, the size of the window and viewport). The values in a .WSP file are initialized by the operator when GFM is running on the 4081.

When creating a file, the operator is encouraged to assign the appropriate GOS standard extension to the filename. Standard extensions facilitate the identification of files and save the operator time when a file is typed. For example, the BATCH utility program assumes a default extension of .BAT. To run a batch file with a .BAT extension, the operator can omit the extension when typing the batch file specifier:

```
#BATCH FD1:VAMPYR
```

The batch file FD1:VAMPYR.BAT is executed.

To run a batch file with an extension other than .BAT, the operator must type the extension as part of the batch file specifier:

```
#BATCH FD1:NOBAT.EXT
```

The batch file FD1:NOBAT.EXT is executed.

It is possible to assign a filename a null extension. A null extension consists of three blank characters. To specify a file with a null extension, type,

```
[device:]filename.
```

The period that follows the filename is required. If the period is not typed, then a default filename extension might be assumed.

For example, an operator wants to display the contents of the ASCII file DK1:NULEXT. on the screen by running the TYPE utility program. DK1:NULEXT. is assigned a null extension. By omitting the period after the filename when typing,

```
#TYPE DK1:NULEXT
```

the TYPE program assumes the default filename extension .ASM and attempts to display the contents of the file DK1:NULEXT.ASM.

By including the period after the filename when typing,

```
#TYPE DK1:NULEXT.
```

the TYPE program displays the contents of the file DK1:NULEXT.(no default filename extension is assumed).

Specifying Files Within Library Files

A library file is a file in a file structure that is itself capable of containing a number of files. A nested file is a file that is created within a library file. (For a detailed explanation of library files, see **Formatting A Library File** in this section.)

A file nested within a library file on a formatted medium is listed in the library file's directory. The library file itself is listed in the primary directory (the formatted medium's directory). Therefore, to locate a file nested within a library file, GOS requires the name of the nested file, the name of the library file, and the name of the device in which the formatted medium is inserted.

To specify a nested file, type,

```
[device:]library filename[.extension]/filename[.extension]
```

device specifies the device in which the formatted medium containing the nested file is inserted.

library filename specifies the library file that contains the nested file. If the library filename extension is .LIB, the GOS standard extension for a library file, then it can be omitted when typing the library file specifier. If no extension is typed, GOS assumes a default extension of .LIB for a file specifier that precedes the /. Any library filename extension other than .LIB must be typed.

/ is the GOS library identifier. The file specified after the / is a member of the library file that precedes the /.

filename specifies the nested file. The extension field is optional. In many cases when the extension is omitted from a file specifier, a default extension is assumed. The default extension is determined by GOS or the program that processes the file.

For example, assume a nested file PYNCHN.TOM is a member of the library file PUBLIC.LIB and that PUBLIC.LIB resides on the flexible disc inserted in the second Flexible Disc Unit drive (FD1:). In this case, to specify the file PYNCHN.TOM, type,

```
FD1:PUBLIC/PYNCHN.TOM
```

The GOS library identifier / also allows the operator to specify a series of nested files (for example, a file that is a member of a library file that is a member of another library file). Type,

```
[device:]1st library filename[.ext]/2nd library filename[.ext]/.../nth library  
filename[.ext]/filename[.ext]
```

The nested file specified by **filename** is a member of the nth library file. The nth library file is a member of the nth-1 library file, and so on. All nested library files in the series must be specified.

If the extension field for a library file specifier that precedes a / is omitted, a .LIB extension is assumed.

For example, assume a nested file PEA.POD is a member of the library file SHELL2.LIB; SHELL2.LIB is a member of the library file SHELL1.LIB; and, SHELL1.LIB resides on the flexible disc inserted in FD1:. In this case, to specify the file PEA.POD, type,

FD1:SHELL1/SHELL2/PEA.POD

DEVICE SPECIFIERS

Under GOS, the transfer of file information is not restricted to file-structured devices. For example, the operator can have the contents of a file displayed on the Display Monitor or drawn on the Plotter. Or, the operator can transfer data from the Graphics Tablet or keyboard and store it in a file.

For data transfer operations, GOS allows the interchange of devices that share the same general characteristics. This feature is called device independence. (Device independence is explained in more detail under **File Processing** in this section.) The operator need only specify the device that sends the data and the device that receives the data. GOS then takes the steps necessary to ensure compatibility between the devices.

As a convenience to the operator, each device is represented by a 2-3 character mnemonic. To specify a device, the operator types the device mnemonic. The device mnemonic must end with a colon (:).

Table 8-2 lists the GOS device mnemonics for all valid 4081 devices.

Table 8-2
SYSTEM DEVICES

Device	Mnemonic	Description
Keyboard	KB:	Valid for ASCII input/output (output sent to Display Monitor)
Display Monitor	DC:(DC0:) DC1:-DC3:	Valid for ASCII and graphic output
Joystick	JOY:	Valid for graphic input (locator mode only)
Communications Interface	CM:(CM0:) CM1:-CM5:	Valid for transmitting data to and receiving data from a host computer or external device
Line Printer	LP:(LP0:) LP1	Valid for ASCII output
Graphics Tablet	TB:(TB0:) TB1:	Valid for graphic input (locator and tracer modes)
Plotter	PL:(PL0:)	Valid for ASCII and graphic output and graphic input (locator mode only)
Cartridge Tape Unit	CT:(CT0:) CT1:	Valid file structured device
Flexible Disc Unit	FD:(FD0:) FD1:-FD3:	Valid file structured device
Hard Disc Unit	DK:(DK0:) DK1:-DK7:	Valid file structured device
System Device	SYS:	Default file structured device for system utility programs
User Device	USR:	Default file structured device for user files
Graphic Input Device	GIN:	Default graphic input device
Null Device	NUL:	Discards output; returns EF (End of File) error on input

In Table 8-2, the first mnemonic listed for each device represents the first device or, for file-structured devices, the first device drive. The other mnemonics listed for each device represent additional devices or additional device drives. For example, if a 4081 has two Line Printers, then the first of the two Line Printers is LP: (or LP0:). The second Line Printer is LP1:.

The brief description of each device indicates the data transfer operations for which the device is valid. A device that is valid for output operations can receive data. A device that is valid for input operations can transmit data. For example, the Line Printer is valid for ASCII output. This means that the contents of an ASCII file can be printed on the Line Printer. The Line Printer, however, is not valid for input operations; therefore, the Line Printer cannot transmit data to another device.

The mnemonic GIN: (Graphic Input) is similar to the SYS: and USR: mnemonics. The GIN: mnemonic is assigned to the device generally used for graphic input operations (either the Joystick, Graphics Tablet, or Plotter). The GIN: mnemonic, like SYS: and USR:, is automatically assigned to a device when the GOS program is loaded into Memory. The device assignment is determined by the user when the GOS IPL tape is generated. The operator, however, can reassign the GIN: mnemonic to a different device by running the SET utility program.

The Null Device (NUL:) is generally used by a programmer as a convenient way to get rid of unnecessary data. Data transmitted to the Null Device is discarded and unrecoverable.

FORMATTING A MASS STORAGE MEDIUM

FORMATTING PROCEDURE

A blank mass storage medium must first be prepared for the storage of files. The process of preparation is called formatting the medium. Formatting ensures that the structure of a medium is compatible with GOS. All GOS file functions are based on a standard structured medium. GOS can process files only on a medium that is structured according to the standard.

The GOS utility program FORMAT allows the operator to format a medium according to the GOS standard. A formatted medium is called a file structure because it has the capacity to contain a number of files that are accessible by name. The mass storage device in which the medium is inserted is called a file-structured device. For example, a formatted flexible disc is a file structure; the Flexible Disc Unit is a file-structured device.

The following procedure describes how to format a medium on the 4081:

1. Ensure that the 4081 is turned on and working properly, the GOS program is loaded into memory, the system disc or tape is inserted in the appropriate device drive, and all peripheral devices to be used are connected to the 4081 and turned on.
2. Insert the medium to be formatted in the appropriate device drive. If there is more than one device drive, the medium can be inserted in any one of them. Ensure that the medium and the drive are not write-protected. (For a detailed explanation of how to write-protect each file-structured device, see **File-Structured Devices** in this section.)

Example

Insert a flexible disc in the second Flexible Disc Unit drive(FD1:). Assume that the system disc is already inserted in FD0: and that FD0: is assigned the mnemonic SYS:.

3. Ensure that the 4081 is in Command mode and ready to accept characters typed on the keyboard. These conditions are indicated, generally, when the prompt character # is displayed on the screen and the first prompt light (labeled REQUEST INPUT) is lit. If you're not sure what mode you're in, press the SHIFT-ESC keys to enter Command mode.
4. Run the FORMAT utility program by typing,

```
#FORMAT device;N:name
```

then press the RETURN key.

device specifies the device drive in which the medium to be formatted is inserted. **;N** is a switch indicating to the FORMAT program that a structure identifier is specified. The structure identifier is a name of 1-6 characters that is intended only as an operator's aid for further identifying a file structure. The structure identifier is listed in the directory of a file structure when the DIR utility program is run. The structure identifier is ignored by GOS and is not part of the standard file specifier. The structure identifier is separated from the ;N switch by a colon(:).

After pressing the RETURN key, the following message is displayed:

Mount volume to be formatted
Type a "G" to Go

Example

Type,

#FORMAT FD1::N:MYDISC

then press the RETURN key.

FD1: specifies the flexible disc in the second Flexible Disc Unit drive (the medium to be formatted). **MYDISC** is the structure identifier. The following message is displayed:

Mount volume to be formatted

Type a "G" to Go

5. Press the G key then press the RETURN key. The following message is displayed:

Formatting in progress

When the formatting is completed, the message **Formatting done** is displayed. A message also informs the operator of the number of bad blocks, if any, on the medium. A bad block is a section of the medium that has been tested and found defective by the FORMAT program. The bad block is automatically assigned read and write-protect attributes so that data cannot be stored on or retrieved from a bad block.

The operator can now create and store files on the formatted medium.

Example

If the flexible disc is properly inserted in FD1: (the "volume" to be formatted is "mounted"), press the G key then press the RETURN key. The following message is displayed:

Formatting in progress

When the formatting is completed, the following message is displayed:

0 Bad Blocks**Formatting done**

The flexible disc in FD1: is now formatted. The disc contains no bad blocks.

To view a displayed listing of the new file structure's directory, type,

#DIR FD1:

The following is displayed:

```
Directory Structure of FD1 MYDISC 2-May-77
Directory Blocks Available- 8, Used- 1
```

```
Files-0 ,Blocks-0 ,Free-1224 ,Largest-1224
```

The formatted flexible disc's directory includes such information as the structure identifier(**MYDISC**), the date that the disc was formatted(**2-May-77**), and the number of blocks of available empty space on the disc(**Free=1224**).

STRUCTURE OF THE FORMATTED MEDIUM

The FORMAT program prepares a mass storage medium for the storage of files by organizing the medium into a standard format. The standard format allows the 4081 to create and access files on the medium.

When the FORMAT program is run by typing,

#FORMAT device;N:name

it first requests that the operator insert the medium to be formatted in the specified device. FORMAT displays the message:

Mount volume to be formatted
Type a "G" to Go

When the operator presses the G key and then the RETURN key, the FORMAT program displays the message:

Formatting in progress

Now FORMAT determines if a medium is inserted in the specified device. If no medium is inserted, it displays:

Please mount device
Pause
#

The operator can now insert the medium, type CON (the resident command CONTINUE), and press the RETURN key to continue the FORMAT program.

After verifying that a medium is properly inserted in the specified device, the FORMAT program begins formatting the medium. It first divides the medium into equal-sized blocks. A block consists of an area on the medium that is 256 bytes in length (a byte is a group of 8 consecutive binary digits or bits). A few extra bytes containing GOS-related information (for example, the number of the block) are also added to each block. It is simpler, however, for the operator to view each block as 256 bytes of data storage (see Fig. 8-25).

The FORMAT program next verifies the condition of the medium by storing test data on the medium then attempting to read back the stored data. FORMAT compares the data it read with the data it stored. If FORMAT, after several attempts, determines that some of the data read is inaccurate (the data read does not match the data stored), then the area of the medium on which the inaccurate data was stored is assumed defective. The block or blocks comprising the defective area are labeled bad blocks (see Fig. 8-26).

Bad blocks are automatically assigned write and read-protect attributes; therefore, data cannot be stored on or retrieved from a bad block. A bad block is listed in the directory of a file structure as **BLKxxx.BAD**. **xxx** represents the number of the block on the medium in hexadecimal notation. For example, if the seventh block on the medium is bad, the block is listed in the directory as **BLK007.BAD**. If several consecutive blocks are bad, only the first

block number is listed followed by the size in blocks of the defective area. For example, if ten consecutive blocks are bad starting from the twelfth block on the medium, the bad blocks are listed in the directory as **BLK00C.BAD [10]** (C represents the decimal number 12 in hexadecimal notation).

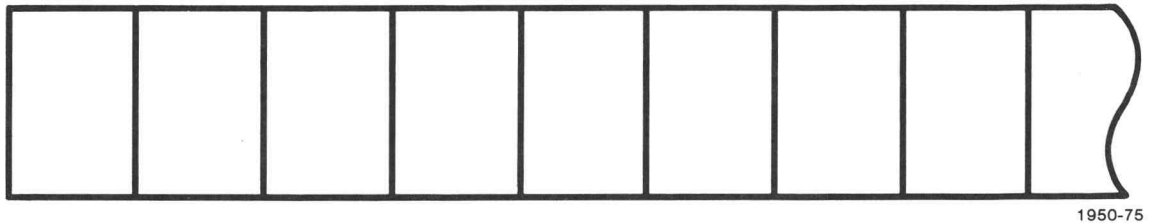


Fig. 8-25. The mass storage medium is first divided into equal-sized blocks of 256 bytes each.

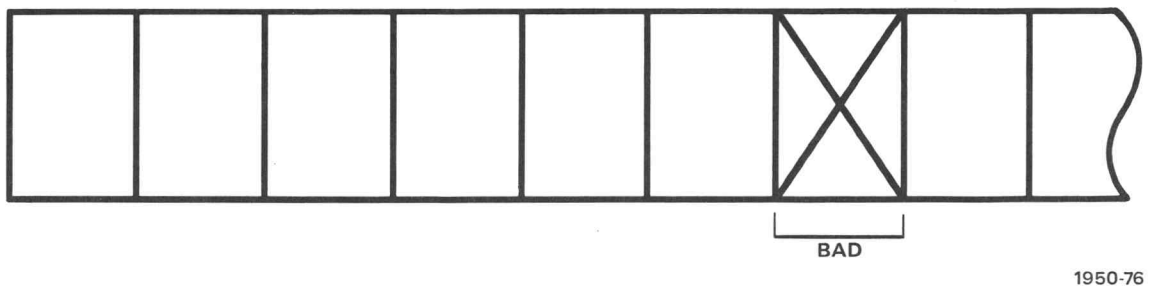


Fig. 8-26. Blocks that are found defective are labeled bad blocks and are automatically read and write-protected.

To display a listing of the bad blocks in a file structure, run the DIR utility program using the ;B switch. Type,

```
#DIR device;B
```

device specifies the device drive in which the formatted medium is inserted. **;B** is a switch requesting the DIR program to include a list of the bad blocks in a file structure's directory.

After the FORMAT program has divided the medium into blocks and tested each block for defects, it creates a directory. The directory contains a list of the files in the file structure and the information needed to locate each file. FORMAT creates a directory by first

reserving enough blocks at the beginning of the medium to contain the specified or default number of directory entries (each file listed in the directory is called an entry). The number of directory entries can be specified when running the FORMAT program using the ;E switch. Type,

#FORMAT device;N:name;E:n

n is the number of entries to be reserved in the file structure's directory. If no number is specified (the ;E switch is omitted), then a default number of directory entries is assumed. The default number depends on the medium to be formatted. Table 8-3 shows the default number of directory entries for each mass storage medium:

Table 8-3

DEFAULT DIRECTORY ENTRIES

Medium	Number of Directory Entries (default)
Hard Disc	720
Flexible Disc	120
Tape Cartridge	120

To reserve enough blocks to contain all directory entries, FORMAT requires not only the number of directory entries but also the amount of space each entry occupies. Generally, each directory block contains a maximum of 15 entries (each entry occupies 16 bytes). The operator, however, can reserve extra space for each directory entry by using the FORMAT program switch ;X. Type,

#FORMAT device;N:name;X:n

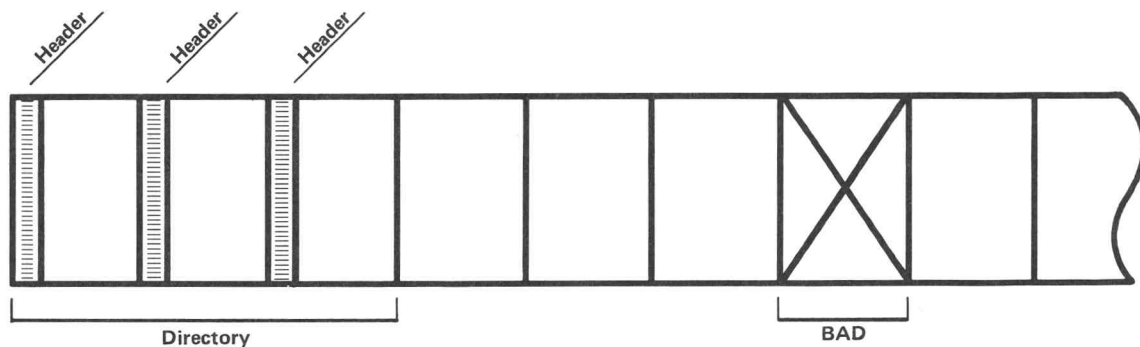
n is the number of extra bytes to be reserved for each directory entry (**n** must be an even integer). If the ;X switch is omitted, no extra bytes are reserved for a directory entry.

Extra bytes in a directory entry are usually reserved for additional information about a file that is not provided by GOS. For example, a programmer may want to reserve 2 extra bytes per directory entry to store the initials of the person who created the file. (See the Plot 80: GOS Programmer's Reference for information on programming file structures.)

Given the number of directory entries desired and the amount of space each entry occupies, FORMAT determines the number of blocks to reserve for a file structure's directory. If the operator does not specify the number of directory entries and does not

reserve any extra bytes (the ;E and ;X switches are omitted when running the FORMAT program), then FORMAT assumes that each directory block contains a maximum of 15 entries. Therefore, the flexible disc and tape cartridge, both with a default number of 120 directory entries, require 8 directory blocks. The hard disc, with a default number of 720 directory entries, requires 48 directory blocks.

After reserving the required number of directory blocks, FORMAT creates a header at the beginning of each directory block. The header contains the bookkeeping information for the directory. It includes the number of blocks in the directory, the number of extra bytes reserved for each entry, the structure identifier, and the date the medium was last formatted. The header occupies the first 16 bytes of each directory block (see Fig. 8-27).



1950-77

Fig. 8-27. Directory blocks are reserved at the beginning of the formatted medium. A header is created in each directory block.

The formatting of the medium is now completed. The FORMAT program displays the message:

```
n Bad Blocks
Formatting Done
```

n is the number of defective blocks on the medium.

FORMATTING A LIBRARY FILE

FORMATTING PROCEDURE

A library file is a file in a file structure that is itself capable of containing a number of files. It can be viewed as a file structure within a file structure.

A library file is created by formatting a portion of a formatted mass storage medium. As when formatting a mass storage medium, the operator runs the GOS utility program FORMAT to format a library file.

The following procedure describes how to format a library file on the 4081:

1. Ensure that the 4081 is turned on and working properly, the GOS program is loaded into memory, the system disc or tape is inserted in the appropriate device drive, and all peripheral devices to be used are connected to the 4081 and turned on.
2. Insert a medium that has been previously formatted in the appropriate device drive. The library file will be formatted on this medium. Ensure that the medium and the drive are not write-protected. (For write-protect information, see **File-Structured Devices** in this section.)

Example

Insert a flexible disc that has been previously formatted in the second Flexible Disc Unit drive (FD1:). Assume that the system disc is already inserted in FDO: and that FDO: is assigned the mnemonic SYS:.

3. Ensure that the 4081 is in Command mode and ready to accept characters typed on the keyboard. These conditions are indicated, generally, when the prompt character **#** is displayed on the screen and the first prompt light (labeled REQUEST INPUT) is lit. If you're not sure what mode you're in, press the SHIFT-ESC keys to enter Command mode.
4. Run the FORMAT utility program by typing,

```
#FORMAT device:filename[n]
```

then press the RETURN key.

device specifies the device drive in which the formatted medium is inserted. **filename** specifies the 1-6 character name of the library file. Generally, a filename is followed by a 1-3 character extension (the extension is preceded by a period). If no extension is specified, however, FORMAT automatically assigns the filename an extension of .LIB, the GOS standard extension for a library file. **n** is an integer that specifies the desired size of the library file in blocks. (A formatted mass storage medium is divided into equal-sized blocks, each capable of containing 256 bytes of data.) The size of the library file cannot be greater than the largest number of adjacent empty blocks available on the formatted medium (run the DIR utility program for a listing of the largest number of adjacent empty blocks available in a file structure). **n** must be typed enclosed in square brackets; for example, [50].

NOTE

The ;N switch for specifying a 1-6 character structure identifier, though required when formatting a mass storage medium, is optional when formatting a library file. If the ;N switch is omitted, FORMAT automatically assigns the library filename as the structure identifier. The structure identifier is listed in the directory of a library file.

After pressing the RETURN key, the library file is formatted. When the formatting is completed, the message **Formatting done** is displayed. Generally, the formatting of a library file takes much less time than the formatting of a mass storage medium.

The operator can now create and store files in the library file.

Example

To create a library file that is 50 blocks in size on the formatted flexible disc inserted in FD1:, first ensure that there are 50 adjacent empty blocks available on the disc. Display a listing of the number of adjacent empty blocks available by running the DIR program. Type,

#DIR FD1:

then press the RETURN key.

The following is displayed on the screen:

```
Directory Structure of FD1:MYDISC 2-May-77
Directory Blocks Available- 8, Used- 1
```

```
Files-0,Blocks-0,Free-1224,Largest-1224
```

The information **Largest=1224** indicates that there are 1224 adjacent empty blocks available on the medium, plenty of space to format a library file of 50 blocks.

To format the library file, type,

#FORMAT FD1:PUBLIC[50]

then press the RETURN key.

Example (cont)

FD1: specifies the formatted disc inserted in the second flexible disc drive.

PUBLIC specifies the name of the library file to be formatted. Since no filename extension is specified, FORMAT automatically assigns the extension .LIB to the library file. The library file will be listed in the disc's directory as PUBLIC.LIB. **[50]** is the number of blocks reserved for the library file. Since no structure identifier is specified (the ;N switch is omitted), FORMAT assigns the filename PUBLIC as the library file's structure identifier.

When the formatting is completed, the following message is displayed:

Formatting done

To view the results of the formatting, first type,

```
#DIR FD1:
```

then press the RETURN key.

The following is displayed:

```
Directory Structure of FD1 MYDISC 2-May-77
Directory Blocks Available- 8, Used- 1
```

```
PUBLIC LIB [50] 2-May-77 11 08 L
```

```
Files-1,Blocks-50,Free-1174,Largest-1174
```

The new library file is listed as **PUBLIC LIB** in the flexible disc's directory. The library file's size in blocks (**[50]**) and the date and time it was formatted are also listed. The **L** to the far right of the entry is an attribute that indicates the entry is a library file. The FORMAT program automatically assigns the L attribute to every library file formatted.

Now type,

```
#DIR FD1:PUBLIC.LIB
```

then press the RETURN key.

The following is displayed:

```
Directory Structure of FD1 PUBLIC 2-May-77
Directory Blocks Available- 4, Used- 1
```

```
Files-0,Blocks-0,Free-46,Largest-46
```


Example (cont)

The library file's directory contains the structure identifier **PUBLIC** and the date the library file was formatted. The library file contains 46 empty blocks available for the storage of files (**Free=46**). There are 46 empty blocks available out of the 50 blocks reserved for the library file because 4 blocks were automatically reserved by the FORMAT program for the library file's directory (**Directory Blocks: Available = 4**).

STRUCTURE OF THE LIBRARY FILE

The FORMAT program creates a library file on a formatted mass storage medium by organizing the library file into a standard format. The standard format allows the 4081 to create and access files within the library file.

The format of a library file is the same as the format of a formatted mass storage medium. The major difference is that a formatted mass storage medium is a file structure; a library file is both a file and a file structure. Under GOS, a library file is treated as a file (for example, the operator can attribute, copy, delete, and rename it) and a library file is also treated as a file structure (for example, the operator can reformat it, add files to it, and get a listing of its directory).

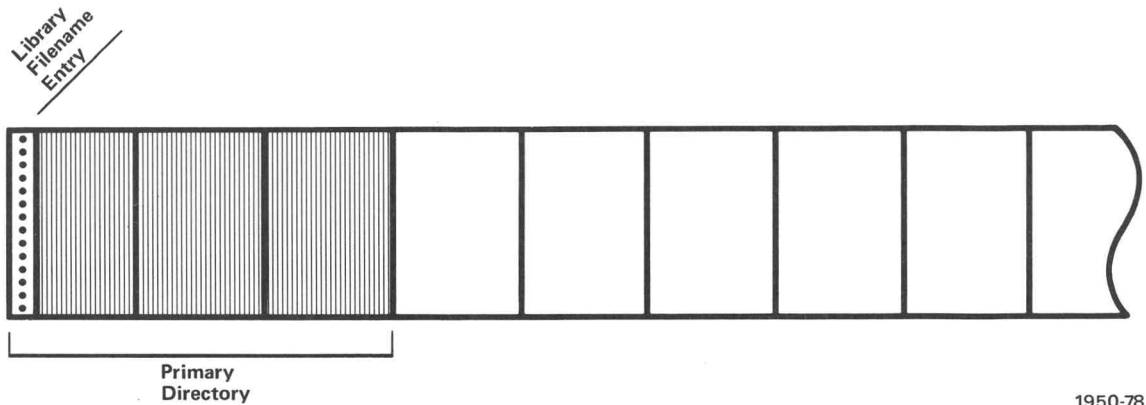
When formatting a new library file, the FORMAT program requires at least 3 arguments: the device in which the formatted mass storage medium is inserted, a 1-6 character library filename, and the size of the library file in blocks. The operator types,

#FORMAT device:filename[n]

then presses the RETURN key.

n is an integer that specifies the library file's size in blocks.

The FORMAT program first creates an entry for the new library file in the directory of the formatted medium (see Fig. 8-28). The formatted medium's directory is called the primary directory to distinguish it from any library file directories on the medium. The library file entry contains the specified library filename (of 1-6 characters) and extension (of 1-3 characters). If no extension is specified, FORMAT automatically assigns the extension **.LIB** to the library filename. (**.LIB** is the GOS standard extension for a library filename.) The entry also contains the size of the library file in blocks, the date and time the library file was formatted, and an **L** attribute. FORMAT automatically assigns the **L** attribute to the entry to indicate a library file.

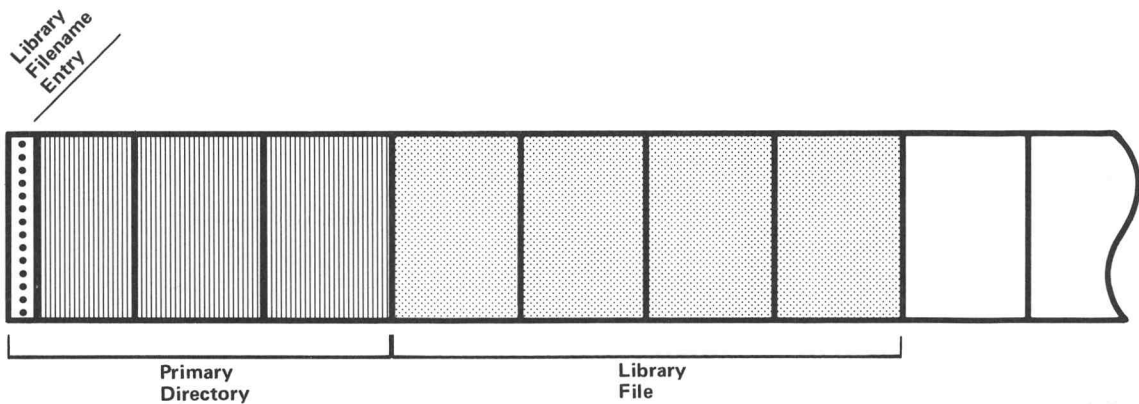


1950-78

Fig. 8-28. An entry for the new library file is created in the primary directory.

The FORMAT program next reserves the specified number of blocks on the medium for the library file (see Fig. 8-29). FORMAT scans the medium for the first set of adjacent empty blocks large enough to contain the library file. If the specified size of the library file is larger than the number of adjacent empty blocks available, the following error message is displayed:

Str Full!



1950-79

Fig. 8-29. The specified number of blocks are reserved for the library file.

To correct the error condition, the operator can either run the GOS utility program SQUISH (which groups all the empty blocks together at the end of the medium), delete unneeded files, or specify a smaller number of blocks for the library file size when running the FORMAT program again.

After reserving the blocks required for the library file, the FORMAT program creates a directory within the library file. The library file directory contains a list of the files in the library and the information needed to locate each file. FORMAT creates a directory by first reserving enough blocks at the beginning of the library file to contain the specified number of directory entries (each file listed in the directory is called an entry). The number of directory entries can be specified when running the FORMAT program using the ;E switch. Type,

#FORMAT device:filename[size];E:n

n is the number of entries to be reserved in the library file's directory. If no number is specified (the ;E switch is omitted), then a default number of directory entries is assumed. The default number of directory entries for a library file is 60.

To reserve enough blocks to contain all directory entries, FORMAT requires not only the number of directory entries but also the amount of space each entry occupies. Generally, each directory block contains a maximum of 15 entries (each entry occupies 16 bytes). The operator, however, can reserve extra space for each directory entry by using the FORMAT program switch ;X. Type,

#FORMAT device:filename[size];X:n

n is the number of extra bytes to be reserved for each directory entry (**n** must be an even integer). If the ;X switch is omitted, no extra bytes are reserved for a directory entry.

Extra bytes in a directory entry are usually reserved for additional information about a file that is not provided by GOS. For example, a programmer might want to reserve a few extra bytes per directory entry to store the astrological birth sign of the person who created the file. (See the Plot 80: GOS Programmer's Reference for information on programming file structures.)

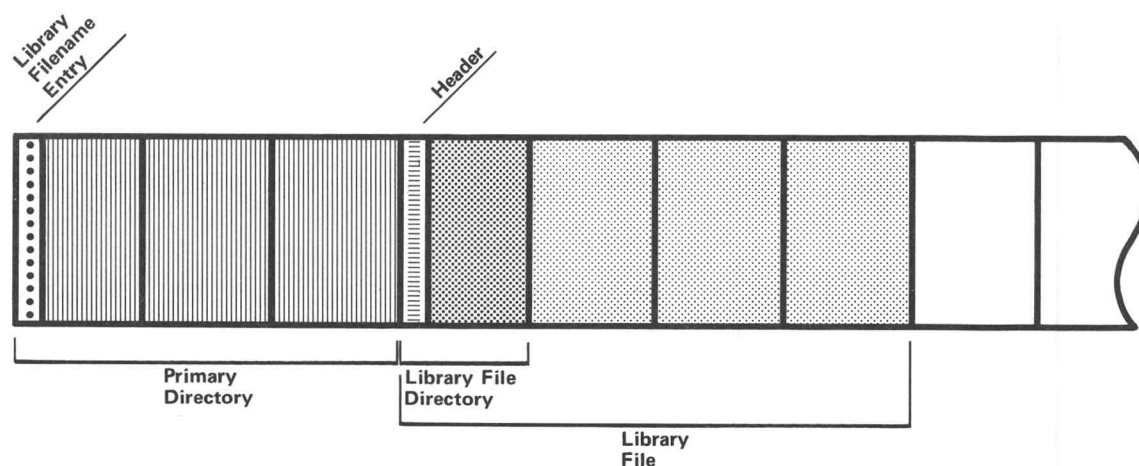
Given the number of directory entries and the amount of space each entry occupies, FORMAT determines the number of blocks to reserve for a library file's directory. If the operator does not specify the number of directory entries and does not reserve any extra bytes (the ;E and ;X switches are omitted when running the FORMAT program), then FORMAT assumes that each directory block contains a maximum of 15 entries. Therefore, a library file with a default number of 60 directory entries requires 4 directory blocks.

If the library file is not large enough to contain the required number of directory blocks (generally, when the specified size of the file is less than or equal to four blocks), then the following message is displayed:

More dir entries than space available!

In this case, specify a larger number of blocks for the library file's size or reduce the number of directory entries specified when running the FORMAT program again.

After reserving the required number of directory blocks, FORMAT creates a header at the beginning of each directory block in the library file (see Fig. 8-30). The header contains the bookkeeping information for the directory. It includes the number of blocks in the directory, the number of extra bytes reserved for each entry, the library file's structure identifier, and the date the library file was last formatted. The header occupies the first 16 bytes of each directory block.



1950-80

Fig. 8-30. Directory blocks are reserved at the beginning of the library file. A header is created in each directory block.

The formatting of the library file is now completed. The FORMAT program displays the message:

Formatting done

FORMATTING LIBRARY FILES WITHIN LIBRARY FILES

Formatting a library file on a formatted mass storage medium creates a file structure within a file structure. GOS also allows the formatting of library files within other library files. Therefore, the operator can create file structures within file structures within file structures, and so on indefinitely as long as there is enough empty space available on the medium.

The process of creating files within other files is called nesting files. The nesting of library files is similar to the stacking of Chinese boxes, one box fitting within a larger box that fits within a still larger box.

To format a library file within another library file on a formatted mass storage medium, type,

#FORMAT device:1st library filename/2nd library filename[n]

then press the RETURN key.

Both the mass storage medium inserted in the specified device and the first library file must have been previously formatted. The character / is a GOS library identifier. It indicates that the file following the / is a member of the library file preceding the /. Therefore, the second library file, the library file to be formatted, will become a member of the first library file.

n is an integer that specifies the desired size of the second library file in blocks. The size of the second library file cannot be greater than the largest number of adjacent empty blocks available in the first library file (run the DIR utility program for a listing of the largest number of adjacent empty blocks available in a file structure). **n** must be typed enclosed in square brackets; for example, [25].

The FORMAT program formats the second library file in the same way it formatted the first library file. FORMAT, however, listed the first library file as an entry in the primary directory (the directory of the formatted mass storage medium). FORMAT lists the second library file as an entry in the directory of the first library file.

The FORMAT program includes the same information in the second library filename entry as it did in the entry for the first library filename: the library's 1-6 character filename, 1-3 character extension (the default extension is .LIB), the size of the library file in blocks, the date and time the library file was formatted, and an L attribute (to indicate the entry is a library file).

FORMAT then reserves the specified number of blocks for the second library file within the area of the medium occupied by the first library file. After reserving the required number of blocks, FORMAT creates a directory for the new library file.

The method for creating a directory is identical to the description given in **Structure Of The Library File** in this section. The FORMAT program reserves enough blocks at the beginning of the library file to contain the specified number of directory entries. The operator specifies the number of directory entries by using the ;E switch when running the FORMAT program. Type,

#FORMAT device:1st library filename/2nd library filename[size];E:n

n is the number of entries to be reserved in the library file's directory. If the number of directory entries is not specified (the ;E switch is omitted), FORMAT assumes a default number of 60 directory entries for the library file.

The operator can also specify the number of extra bytes to reserve for each directory entry by using the ;X switch. Type,

#FORMAT device:1st library filename/2nd library filename[size];X:n

n is the number of extra bytes to be reserved for each directory entry. If the ;X switch is omitted, no extra bytes are reserved for a directory entry (each entry occupies 16 bytes).

Given the number of directory entries and the amount of space each entry occupies, FORMAT determines the number of blocks to reserve for the new library file's directory. If the operator does not specify the number of directory entries and does not reserve any extra bytes (the ;E and ;X switches are omitted when running the FORMAT program), then FORMAT assumes that each directory block contains a maximum of 15 entries. Therefore, a library file with a default number of 60 directory entries requires 4 directory blocks.

If the library file is not large enough to contain the required number of directory blocks (generally, when the specified size of the file is less than or equal to four blocks), then the following message is displayed:

More dir entries than space available!

In this case, specify a larger number of blocks for the new library file's size or reduce the number of directory entries specified when running the FORMAT program again.

After reserving the required number of directory blocks, FORMAT creates a header at the beginning of each directory block in the new library file. The header contains the bookkeeping information for the directory. It includes the number of blocks in the directory, the number of extra bytes reserved for each entry, the library file's structure identifier, and the date the library file was last formatted. The header occupies the first 16 bytes of each directory block.

The formatting of the new library file is now completed. The FORMAT program displays the message:

Formatting done

Example

Assume that the flexible disc inserted in the second Flexible Disc Unit drive (FD1:) has been previously formatted and that a library file with the filename PUBLIC.LIB and a size of 50 blocks has been created on the disc (see the previous examples in this section). Assume that the system disc is inserted in FDO: and that FDO: is assigned the mnemonic SYS:.

To create a library file within the library file PUBLIC.LIB, type,

```
#FORMAT FD1:PUBLIC/LIB2[25]
```

then press the RETURN key.

FD1: specifies the formatted disc inserted in the second flexible disc drive.
PUBLIC specifies the name of the first library file (.LIB is the default extension for a filename preceding the GOS library identifier /.) / is the GOS library identifier. The filename after the / will become a member of the library file preceding the /.

LIB2 specifies the name of the library file to be formatted. Since no filename extension is specified, FORMAT automatically assigns the extension .LIB to the library file. The new library file will be listed in the directory of the first library file as LIB2.LIB. **[25]** is the number of blocks reserved for the new library file. Since no structure identifier is specified (the ;N switch is omitted), FORMAT assigns the filename **LIB2** as the new library file's structure identifier.

When the formatting is completed, the following message is displayed:

Formatting done

To view the results of the formatting, first type,

```
#DIR FD1:PUBLIC.LIB
```

then press the RETURN key.

The following is displayed:

Example (cont)

```
Directory Structure of FD1:PUBLIC 2-May-77
Directory Blocks Available- 4, Used- 1
```

```
LIB2 LIB [25] 2-May-77 11:11 L
```

```
Files-1,Blocks-25,Free-21,Largest-21
```

The new library file is listed as **LIB2 LIB** in the directory of the library file **PUBLIC.LIB**. The new library file's size in blocks (**[25]**) and the date and time it was formatted are also listed. The **L** to the far right of the entry is an attribute that indicates the entry is a library file. The **FORMAT** program automatically assigns the **L** attribute to every library file formatted.

Now type,

```
#DIR FD1:PUBLIC/LIB2.LIB
```

then press the **RETURN** key.

The following is displayed:

```
Directory Structure of FD1:LIB2 2-May-77
Directory Blocks Available- 4, Used- 1
```

```
Files-0,Blocks-0,Free-21,Largest-21
```

The new library file's directory contains the structure identifier **LIB2** and the date the library file was formatted. The library file contains 21 empty blocks available for the storage of file (**Free = 21**). There are 21 empty blocks available out of the 25 blocks reserved for the new library file because 4 blocks were automatically reserved by the **FORMAT** program for the library file's directory (**Directory Blocks: Available = 4**).

It is possible to continue to nest library files, to create a library file within a library file. Type,

```
#FORMAT device:1st library filename/2nd library filename/. . ./nth library
filename[size]
```

The library files preceding the nth library file must have been previously formatted.

The method for formatting the nth library file is identical to the previous description of formatting a library file within another library file. The only difference is that the directory entry for the nth library file is listed in the directory of the library file that directly precedes it, the nth-1 library file.

FILE PROCESSING

File processing concerns how, under GOS, a file is created, deleted, accessed, and appended on the 4081. Most of the information on file processing is mainly of interest to a 4081 programmer. An operator, however, might read the information to get a better idea of how to use the GOS resident commands ASSIGN, CLOSE, and LU. The information on file sizing also may be of interest to a 4081 operator.

THE LOGICAL UNIT

A logical unit is an integer from 0-15 that is assigned to a device (or a file on the device). All input/output operations from and to a device are performed through the logical unit to which the device is assigned.

The logical unit acts as a middleman between a device and the 4081 programmer. A 4081 operator, when running a program, generally does not have to be concerned with logical units. The operator can view the connection between a program on the 4081 and an external device as direct.

Once a device is assigned to a logical unit, the programmer communicates with the device by addressing the logical unit. For example, to read a file on a flexible disc and display the data on the Display Monitor, the programmer first assigns the file to a logical unit, then assigns the Display Monitor to a different logical unit. Now, the programmer need only read data from the logical unit assigned to the file and write the data to the logical unit assigned to the Display Monitor.

The programmer does not communicate directly with a device assigned to a logical unit. GOS relieves the programmer of the tedious procedure of addressing a device, of sensing the device status (for example, is the device ready to receive commands, is it busy, is it even connected to the 4081 ?), and of sending commands to the device in a binary code that changes from one device to the next. Through logical units, GOS offers the 4081 programmer device independence. Device independence means the programmer doesn't have to worry about the idiosyncrasies of individual devices. The programmer can treat devices that share the same general characteristics as if they were identical.

For example, a 4081 programmer writes a program that reads an ASCII file on a flexible disc and displays the contents of the file on the Display Monitor. The programmer has an ASCII file on a tape cartridge that he also wants read and displayed on the Display Monitor. To do this, the programmer does not need to write a new program. Both the Flexible Disc and Cartridge Tape Units are file-structured devices. They share the same general characteristics. The programmer can treat both devices as if they were identical. Therefore, the programmer simply modifies the old program by reassigning the logical unit assigned to the file on the flexible disc to the file on the tape cartridge. Now when the program reads data from the logical unit, it reads from the cartridge tape file assigned to the logical unit.

CREATING A FILE

A file is created on the 4081 by first assigning the new file to a logical unit. This is called opening the file. Once the new file is opened, it is listed in the directory of the file structure in which it's created. For example, if the new file is created on a flexible disc, the new file's entry is listed in the primary directory of the disc. If the new file is created in a library file, the new file's entry is listed in the library file's directory.

Initially, a new file's directory entry contains the file's name, extension, and status. A file's status is either tentative, permanent, or empty. When a new file is first opened, its status is tentative. A tentative (or "latent") file is a new file that is open and capable of receiving data. When all the data has been sent to the tentative file and stored, the file must be closed if the tentative file is to become permanent.

Closing a file is a GOS procedure that is initiated by a programmed request or the GOS resident command CLOSE. When a tentative file is closed, GOS determines if the file contains any data. If the file contains data, it becomes a permanent file. A permanent file remains in a file structure until it is deleted. If the file contains no data when it is closed, it becomes an empty file. An empty file is not listed in a file structure's directory. The space in a file structure occupied by an empty file is available for the storage of new files.

If a tentative file is not closed, it is treated like an empty file. For example, a programmer opens a new file by assigning it to a logical unit. The file's status, at this point, is tentative. Then, instead of closing the tentative file, the programmer reassigns the logical unit to another file or device. By reassigning the logical unit, the programmer has, in effect, lost contact with the tentative file. Since the tentative file was not closed, it becomes identical to an empty file. The space in a file structure occupied by the tentative file is now available for the storage of new files.

PROCESSING A PERMANENT FILE

Before reading data from or writing data to a permanent file, the file must first be opened. A file is opened by assigning it to a logical unit. A permanent file can be opened either for input only or for both input and output.

If a permanent file is opened for input only, then the contents of the file can be read but no data can be written to the file. Writing data to a file that is opened only for input causes an illegal function error. If a permanent file is opened for both input and output, then the contents of the file can be read and additional data written to the file.

When a permanent file is opened for input only, the file's contents and status do not change. If the logical unit assigned to the file is reassigned to another file or device before closing the file, then the permanent file still remains permanent and its contents remain unchanged.

When a permanent file is opened for both input and output, then data is written to it, the file must be closed if the new data is to be saved. If the logical unit assigned to the file is reassigned to another file or device before closing the file, then any data written to the file while the file was open is lost. The original contents of the file, however, are unchanged.

If a new file is created with the same filename and extension as a permanent file in the same file structure, the new file, when closed, supersedes the previous file. For example, assume a permanent file GEMINI.ASM resides on a flexible disc inserted in the second Flexible Disc Unit drive (FD1:). A new file, also named GEMINI.ASM, is then created on the disc in FDI:. First, the new file is opened by assigning it to logical unit 2. The status of the new GEMINI.ASM, at this point, is tentative. The status of the old GEMINI.ASM is still permanent. Data is then written to the new file and the file is closed. The new GEMINI.ASM now becomes a permanent file. The old GEMINI.ASM is superseded by the new file; it becomes an empty file. The space in the file structure occupied by the old GEMINI.ASM is available for the storage of new files. If logical unit 2, however, is reassigned to another file or device before the new GEMINI.ASM is closed, then the old GEMINI.ASM remains a permanent file and the new file is treated as if it were empty.

A permanent file can be deleted from a file structure by a programmed request or by running the DELETE utility program. When a permanent file is deleted, the file's status changes to empty. The space in a file structure occupied by an empty file is available for the storage of new files.

Note

By running the DIR utility program, only the permanent files listed in a file structure's directory are displayed on the screen. Files with either tentative or empty status are not displayed.

FILE SIZING

Under GOS, a file occupies one or more consecutive blocks in a file structure. A block is 256 bytes of storage space on a mass storage medium. The size of a file is specified by the number of blocks the file occupies.

When a file is created on the 4081, the size of the file is either specified by the operator or determined automatically by GOS. In most cases, the operator does not need to specify the size of a new file. When the operator does not specify a file's size, GOS automatically determines the exact size of the file by the following procedure:

1. Finds the largest number of adjacent empty blocks in the file structure in which the new file is created.

Example

When running the DIR utility program by typing,

```
#DIR FD0:,FD1:SYSTEM.DIR
```

the file **SYSTEM.DIR** is created on the flexible disc inserted in the second Flexible Disc Unit drive (**FD1:**). The file contains a listing of the directory of the flexible disc inserted in the first Flexible Disc Unit drive (**FD0:**). The keyboard operator has not specified a size for the new file. In this case, GOS automatically determines the file size.

First, GOS finds the largest number of adjacent empty blocks on the flexible disc in **FD1:**. In this example, assume the largest empty space on the disc occupies 100 blocks.

2. Divides the number in half.

Example

GOS then divides the largest number of adjacent empty blocks on the flexible disc in half. The result is 50 blocks.

3. Creates the file using this number as the temporary file size.

Example

Next, the file is opened for creation by assigning it to a logical unit. The file is listed as an entry in the primary directory of the flexible disc in FD1:. The entry contains the filename (SYSTEM), the extension (DIR), the status (tentative), and the temporary size of the file (50 blocks).

4. Removes any unused blocks from the file when the file is closed.

Example

After the directory listing for the flexible disc in FD0: is written to the new file, the file is closed. Assume the directory listing occupies only the first 8 blocks of the 50 blocks reserved for the new file. GOS removes the 42 unused blocks from the file. The unused blocks are considered empty space and are available for the storage of new files.

The size of the file is trimmed from the original 50 blocks to only 8 blocks. (Under GOS, the file size must always be a whole number of blocks. For example, if the directory listing occupied only 7 1/2 blocks of the new file, the file's size would still be trimmed to 8 blocks.) The file SYSTEM.DIR now becomes a permanent file with a file size of 8 blocks.

The operator always has the option of specifying the file size. The operator specifies the size of a file by typing the size of the file in blocks directly after typing the file specifier. The file size must be enclosed in square brackets:

device:filename.extension[n]

n is an integer specifying the size of the file in blocks.

Because of the limitations of the GOS procedure for automatic file sizing, it is sometimes desirable for the operator to specify the file size. If no size is specified when the file to be created is larger than half the largest number of adjacent empty blocks available in a file structure, GOS is unable to create the file. The error message **Str Full!** or **I/O Error EM on LU n** (n is the logical unit number assigned to the file) is displayed. In this situation, the operator should specify the file size.

If the same error message is still displayed, the operator can either run the SQUISH utility program or delete unneeded files in the file structure.

Example

An operator wants to copy a file on the tape cartridge inserted in the first Cartridge Tape Unit drive (CT0:) to the flexible disc inserted in the first Flexible Disc Unit drive (FD0:). The file on the tape cartridge, THEN.OBJ, occupies 30 blocks. The largest number of adjacent empty blocks available on the flexible disc is 50 blocks.

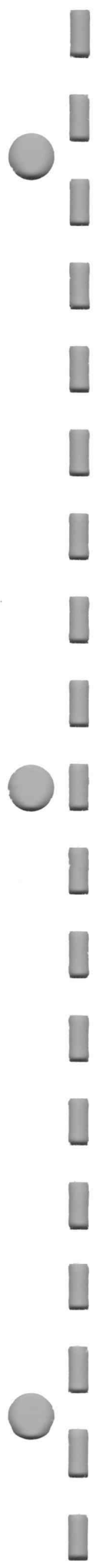
Using the automatic file sizing procedure, GOS would be unable to create a copy of THEN.OBJ on the flexible disc. GOS would divide the largest number of adjacent empty blocks available on the disc, 50 blocks, by two. The result, 25 blocks, would become the temporary size of the new file. Since THEN.OBJ occupies 30 blocks, there would not be enough space available in the new file to copy the entire contents of THEN.OBJ. In this situation, the operator must specify the size of the new file.

The operator specifies the new file's size when running the COPY utility program by typing,

```
#COPY FD0:NOW.OBJ[30]=CT0:THEN.OBJ
```

then pressing the RETURN key.

The new file **NOW.OBJ** is created on the flexible disc in **FD0:**. As specified by the operator, the file occupies **30** blocks. The contents of NOW.OBJ are identical to the contents of THEN.OBJ.



SECTION 9

GOS RESIDENT COMMANDS

CONTENTS

	Page
ASSIGN.....	9-5
CHARSIZE	9-7
CLOSE.....	9-8
CONTINUE.....	9-9
LOAD	9-10
LU	9-12
RUN	9-14
START.....	9-15
EXAMINE (E)	9-17
EXAMINE NEXT HALFWORD (+)	9-18
EXAMINE PREVIOUS HALFWORD (-)	9-19
DEPOSIT (D)	9-20
COMMENT (*)	9-22

Section 9

GOS RESIDENT COMMANDS

The GOS resident commands are stored as part of the GOS program on the GOS IPL tape cartridge. When the GOS program is loaded into the 4081 memory, the resident commands are also loaded into memory. The name "resident command" means that the command is resident in memory when the 4081 is operating. The GOS resident commands include:

- ASSIGN
- CHARSIZE
- CLOSE
- CONTINUE
- LOAD
- LU
- RUN
- START
- EXAMINE (E)
- EXAMINE NEXT HALFWORD (+)
- EXAMINE PREVIOUS HALFWORD (-)
- DEPOSIT (D)
- COMMENT (*)

The 4081 must be operating in Command mode before a resident command can be executed. Generally, when the 4081 is operating in Command mode, the prompt character # is displayed on the screen. (To make sure that the 4081 is operating in Command mode, press the SHIFT-ESC keys. The message <Attn> should appear on the screen followed by the prompt character #). To execute a resident command, type the name of the command and the required arguments, then press the RETURN key or the LF key. The operator can type the command name and arguments using either upper or lower case characters.

NOTE

If a GOS resident command is longer than 3 letters, the operator need type only the first 3 letters of the command.

Table 9-1 explains most of the phrases and symbols used in the examples of GOS resident command syntax in this section.

Table 9-1
GOS RESIDENT COMMAND SYNTAX SYMBOLS

file	A GOS standard file specifier. The GOS standard file specifier has the following format: device:filename.extension
device	A device mnemonic. The GOS device mnemonic consists of 2-3 characters followed by a colon (:).
program argument	A value, device, or file that is operated on by a program.
hex location	A specified location in memory. The address of the memory location is given in hexadecimal notation. Four hexadecimal digits are required to specify a 16-bit memory location (each hexadecimal digit represents four binary digits or bits). For the LOAD, RUN, and the START commands, the hexadecimal location is preceded by the character @.
[]	Information enclosed in brackets is optional.
/	Library file identifier (e.g. the file SYSEQU.ASM in the library SYSLIB.LIB is specified as SYSLIB/SYSEQU.ASM).
:	Device identifier; follows the device mnemonic (e.g. .OBJ).
.	Filename extension identifier; precedes the extension (e.g. .OBJ).
#	GOS command processor input request.

NOTE

In the examples for the resident commands, the characters printed in boldface type represent the information that is actually displayed on the screen by GOS or the resident command. The operator need type only the characters that are not printed in boldface type.

ASSIGN

Syntax

ASSIGN n=file or device

Defaults

for a file, device:=USR:
 .extension=none

for a device, device:=KB:

Description

Assign a logical unit number **n** to a device or file. There are 16 user-assignable units, numbered 0-15. GOS initially assigns all logical units to the keyboard. The LU command is used to display a list of logical unit assignments.

When a file is specified, the default device is USR:. If the file specified does not yet exist, the operator must specify the file size in square brackets after the file name. (See **File Sizing** in the section **Files** for details.)

Example

#ASS 4=DC:

Assign logical unit 4 to the Display Monitor.

#ASS 2=

Assigns logical unit 2 to the keyboard (KB:), the default device.

#ASS 12=NEW.LST[50]

Assign logical unit 12 to the new file USR:NEW.LST. Open and allocate 50 blocks for that file. After data is written to the file and the file is closed, the file NEW.LST becomes a permanent file.

#RUN ROVER

Please assign LU 7 to LP:

Pause

#ASS 7=LP:

#LU

LU	Device	Name	Ext
0	KB		
1	KB		
2	KB		
3	KB		
4	KB		
5	KB		
6	KB		
7	LP		
8	KB		
9	KB		
10	KB		
11	KB		
12	KB		
13	KB		
14	KB		
15	KB		

#CON

The operator runs the program USR:ROVER.OBJ which displays the messages **Please assign LU 7 to LP:** and **Pause** on the screen. The system prompt character # is then displayed. The operator executes the ASSIGN command to assign logical unit 7 to the Line Printer. The LU command is executed to display a list of logical unit assignments (LU 7 is shown assigned to LP). The CON command is then executed to continue execution of the user program.

CHARSIZE

Syntax

CHARSIZE *n*

Defaults

none

Description

Set the character size for the Display Monitor. The size *n* ranges from 1, the smallest character size, to 10, the largest. Initially, GOS assigns the character size 4.

The following table lists, for each of the 10 character sizes, the number of characters that can be displayed on a line, the number of lines that can be displayed on the screen, and the number of characters that can be displayed on the screen. A full monitor viewport (MVP), the entire viewing area of the screen, is assumed.

TABLE 9-2
CHARACTER SIZE LIMITS

Char. Size	Chars. Per Line	Lines Per Full Screen	Chars. Per Full Screen
1	341	173	58,993
2	171	86	14,706
3	114	57	6,498
4	86	43	3,698
5	69	34	2,346
6	57	28	1,596
7	49	24	1,176
8	43	21	903
9	38	19	722
10	35	17	595

Example

#CHA 3

Set the character size to 3. The system prompt # is then displayed in the new character size.

CLOSE

Syntax

CLOSE

Defaults

none

Description

Close all open files, release dynamic memory (refresh graphics cleared), reassign all logical units to the keyboard, and release all user-defined interval control blocks (defined by an SVC 8 instruction or FORTRAN TIMER routine). **Exit** is then displayed on the screen.

Example

Erasing the screen alone does not remove refresh graphics from the Display Monitor. To remove refresh graphics, first type the CLOSE command, press the RETURN key, and then erase the screen.

During program execution, the error message **No Buf!** may appear if no dynamic memory remains. If the contents of dynamic memory are no longer needed, execute the CLOSE command to clear dynamic memory. The CLOSE command also closes all open files.

CONTINUE

Syntax

CONTINUE

Defaults

none

Description

Continue the execution of a suspended program. A running program can be suspended by pressing the SHIFT-ESC keys or by including an SVC 2 Pause or Fortran PAUSE instruction in the program's source code. The 4081 then enters Command mode. At this point, if any programs other than certain GOS resident commands are run, the suspended program cannot be continued. If an attempt is made to continue the suspended program, the character ? is displayed.

The GOS resident commands that can be executed without interfering with a suspended program are CHARSIZE, LU, E, +, -, and *. When executing the resident commands LOAD or D, make sure that the memory locations occupied by the suspended program are not loaded with new data (unless intended). When executing the resident command ASSIGN, make sure that the logical units used by the suspended program are not reassigned to different devices (unless intended). Executing the resident commands CLOSE, RUN, or START prevents the continuation of the suspended program.

Example

```
#RUN JEKYL
```

Please put MYDISK in FD1

Pause

```
#CON
```

The RUN command loads and executes the program USR:JEKYL.OBJ. The messages **Please put MYDISK in FD1** and **Pause** are displayed by the program. After inserting the flexible disc labeled MYDISK in FD1:, the operator enters the CONTINUE command to continue execution of the user program.

LOAD

Syntax

LOAD file [@ hex location]

Defaults

for the file,	device:=USR: .extension=.OBJ
for hex location,	@ hex location=lowest possible user-available location (.GOSTOP)

Description

Load a program load file from a file-structured device into memory at the specified hexadecimal location. If no location is specified, the program is loaded at the lowest possible user-available location (.GOSTOP). When the program is loaded into memory, the command processor displays the following:

Bias xxxx	(address at which the program is loaded into memory)
Start xxxx	(address at which the program starts execution)
End xxxx	(address of the next available memory location after the end of the program)

All addresses are displayed in hexadecimal notation. The LOAD command does not start program execution. The START command can be used to execute the program. See the section **Running a Program** for details on executing a program.

Example

#LOA LOSER@8000

Bias 8000

Start 8000

End A52E

Load the program USR:LOSER.OBJ at memory location X'8000'. The bias and the starting address of the program are X'8000'. The next available memory location is X'A52E'.

#LOA COAL.MYN

Bias 6FD0

Start 6FD0

End 84AD

Load the program USR:COAL.MYN. By default, the program is loaded at the lowest possible user-available location. The bias and starting address of the program are X'6FD0'. The next available location is X'84AD'.

LU

Syntax

LU

Defaults

none

Description

Display a list of the 16 user-assignable logical units along with the name of the device or file assigned to each one. GOS initially assigns all logical units to the keyboard (KB:). The ASSIGN command is used to reassign logical units.

Example

```
#DIR
I/O error DU lu 1
Pause
#LU
```

LU	Device	Name	Ext
0	KB		
1	FD1		
2	KB		
3	KB		
4	KB		
5	KB		
6	KB		
7	KB		
8	KB		
9	KB		
10	KB		
11	KB		
12	KB		
13	KB		
14	KB		
15	KB		

```
#CON
```

While running the DIR utility program, the error message **I/O error DU lu 1** is displayed on the screen. The error message means that the device assigned to LU 1 is not available. The operator enters the LU command and LU 1 is shown assigned to FD1: (the second Flexible Disc Unit drive). The operator then checks the second Flexible Disc Unit drive. The door on FD1: is not closed (the reason for the error message). After closing the door, the operator enters the CONTINUE command to continue execution of the DIR utility program.

RUN

Syntax

RUN file [@ hex location] [program argument]

Defaults

for the file, device:=USR:
 .extension=.OBJ

for hex location, @ hex location=lowest possible
 user-available
 location (.GOSTOP)

Description

Load a program from a file-structured device into memory at the specified hexadecimal location, then execute the program. If no location is specified, the program is loaded at the lowest possible user-available location (.GOSTOP). **program argument** specifies an argument to be transferred to the running program (see the section **Running a Program** for details).

Example

#RUN SYS:DELETE@8000 FUDD.OBJ

Load the program SYS:DELETE.OBJ at location X'8000' and begin execution. Pass the argument file specifier FUDD.OBJ to the program.

#RUN DICK.RUN

Load the program USR:DICK.RUN and begin execution. By default, the program is loaded at the lowest possible user-available location.

START

Syntax

START [@hex location] [program argument]

Defaults

for hex location, @hex location = location of the last
program loaded into
memory

Description

Start execution at the specified memory location. If no location is specified, the starting address of the last program loaded into memory is assumed. **program argument** specifies an argument to be transferred to the running program (see the section **Running a Program** for details). The START command, in most cases, is used with the LOAD command.

Example

```
#LOA TRIAL  
  Bias 974A  
  Start 974A  
  End B448
```

#STA

The LOAD command loads the program USR:TRIAL.OBJ into the 4081 memory. The START command then begins execution of the program.

GOS RESIDENT COMMANDS
START

The START command, when used with the LOAD command, is extremely helpful on a standard 4081 Graphic System with only one Cartridge Tape Unit drive. For example, the operator wants to display a graphics file MAP.PLT stored on a tape cartridge. The DISPLA utility program, however, is stored on the system utilities tape cartridge. With only one Cartridge Tape Unit drive, both tape cartridges cannot be inserted at the same time.

In this case, the operator must first insert the system utilities tape cartridge into the drive, then load the DISPLA utility program into memory:

```
#LOA DISPLA
  Bias 73C0
  Start 73C0
  End 7B36
#
```

Now, the operator removes the system utilities tape cartridge from the drive and inserts the tape cartridge that contains the MAP.PLT file. The operator can then start the execution of the DISPLA utility program with the START command. The graphics file MAP.PLT is specified as the program argument for DISPLA.

```
#STA MAP.PLT
```

The contents of MAP.PLT are now displayed on the screen.

EXAMINE

Syntax

E hex location

Defaults

none

Description

Display the contents of the specified memory location. The contents are displayed in hexadecimal notation. An even (halfword) address should be specified. If an odd address is specified, the halfword at the address minus one is examined (for example, if the odd address X'8001' is specified, then the halfword at the address X'8000' is examined).

Example

#E 8842

FFFF

Displays the contents of memory location X'8842'. The contents are displayed in hexadecimal notation (X'FFFF').

#E C0

0020

Displays the contents of memory location X'00C0' (the location of the GOS Job Status Word, the JSW). The contents of the JSW are displayed in hexadecimal notation (X'0020').

EXAMINE NEXT HALFWORD

Syntax

+

Defaults

none

Description

Examine the contents of the halfword in memory after the last location examined (one halfword is added to the previous location).

Example

#+

CODE

If the last location examined was X'8842', the contents of memory location X'8844' are displayed. The contents are displayed in hexadecimal notation (X'CODE').

EXAMINE PREVIOUS HALFWORD

Syntax

-

Defaults

none

Description

Examine the contents of the halfword in memory before the last location examined (one halfword is subtracted from the previous location).

Example

#-

FFFF

If the last location examined was X'8844', the contents of memory location X'8842' are displayed. The contents are displayed in hexadecimal notation (X'FFFF').

DEPOSIT

Syntax

D hex value

Defaults

none

Description

Deposit the specified hexadecimal value in the last memory location examined.

Example

#D 4081

The contents of the last memory location examined are replaced by X'4081'.

The E, +, -, and D commands can be used to enter data and executable machine code. For example, enter the following commands and arguments (press the RETURN key after each command), then watch the prompt lights in the upper right corner of the Keyboard Unit put on a show. Press the SHIFT-ESC keys to terminate the program. Type,

#E 8000	# +
#D E170	#D E120
# +	# +
#D 0199	#D 8012
# +	# +
#D E120	#D 2208
# +	# +
#D 8012	#D 000F
# +	# +
#D E170	#D 0004
# +	#STA @ 8000
#D 0166	

The hexadecimal numbers specified for the DEPOSIT (D) commands are machine language instructions (assembled assembly language instructions) and data. The EXAMINE (E) command displays the contents of memory location X'8000'. The first DEPOSIT (D) command replaces the contents of location X'8000' with the value specified as the argument (X'E170'). The EXAMINE NEXT LOCATION (+) command then displays the contents of X'8002'. The second DEPOSIT command replaces the contents of location X'8002' with the specified argument (X'0199'). The EXAMINE NEXT LOCATION and DEPOSIT commands alternate until all of the instructions and data have been loaded into memory. The START command then begins execution of the program by specifying the starting memory location X'8000'.

COMMENT

Syntax

*text

Defaults

none

Description

Indicates a comment line. The line of text following the *(asterisk) is ignored by the GOS command processor. In other words, the line of text is displayed on the screen, but GOS does not interpret it as a command or program to be executed.

Example

```
# * This is an example of a comment. This line is ignored.  
#
```

```
#*this hard copy shows the error message displayed  
#*when I tried to run Bob's program  
#  
#run bob  
Dir I/O error DU 1u 16  
#
```

The operator tried to run the program load file FD1:BOB.OBJ and received an error message. The operator types a comment on the screen explaining the situation, then makes a hard copy of the screen to document the error.

SECTION 10

RUNNING A PROGRAM

CONTENTS

	Page
Program Syntax	10-5
Concise Command Language (CCL)	10-7
Wild Cards	10-8

Section 10

RUNNING A PROGRAM

Programs that are executed by the 4081 are generally written in either the Plot 80: Assembly Language or Plot 80: FORTRAN IV language. An assembly language or FORTRAN source program file cannot be executed. The assembly language or FORTRAN source program file must first be assembled or compiled. The assembly or compiling process creates a program load file (or object file) that is stored on a medium (disc or tape) inserted in a file-structured device connected to the 4081. The program load file can then be executed. (Program load files generally have the GOS standard filename extension .OBJ.)

Two steps are required to run a program load file stored on a medium inserted in a file-structured device connected to the 4081:

- The program load file must be transferred from the file-structured device into the 4081 memory. This step is called loading. Usually program load files have the extension .OBJ.
- The program can then be executed, generally starting with the first statement of the program.

Table 10-1 explains most of the phrases and symbols used in the examples of program syntax in this section.

Table 10-1
GENERAL PROGRAM SYNTAX SYMBOLS

file	A GOS standard file specifier. The GOS standard file specifier has the following format: device:filename.extension
device	A device mnemonic. The GOS device mnemonic consists of 2-3 characters followed by a colon(:).
program argument	A value, device, or file that is operated on by a program.
[]	Information enclosed in brackets is optional.
/	Library file identifier (e.g. the file SYSEQU.ASM in the library SYSLIB.LIB is specified as SYSLIB/SYSEQU.ASM).
:	1. Device identifier; follows the device mnemonic (e.g. FD0:). 2. Separates a switch from its value (e.g. ;N:NAME).
;	Switch identifier; precedes the switch (e.g. ;H).
.	Filename extension identifier; precedes the extension (e.g. .OBJ).
#	GOS command processor input request.
,\$,*	General purpose input request (input depends on the program running).

NOTE

In the examples for running programs, the characters printed in boldface type represent the information that is actually displayed on the screen by GOS or the program running. The operator need type only the characters that are not printed in boldface type.

Program Syntax

The phrase “running a program” means loading and executing a program load file. A program stored on a medium inserted in a file-structured device connected to the 4081 can be run in three ways:

By typing,

```
#file [@hex location] [ program argument]
```

This format is generally used to run system programs, such as the GOS utility programs. The default device is SYS:. The default extension is .OBJ.

Example

```
#DIR
```

Loads and executes the GOS utility program load file SYS:DIR.OBJ.

by typing,

```
#RUN file [@hex location] [ program argument]
```

This format is often used to run user programs. The default device isUSR:. The default extension is .OBJ. The RUN command performs the same procedure as the LOAD and START command sequence.

Example

```
#RUN BLAZER @8000
```

Loads and executes the user program USR:BLAZER.OBJ. The program is loaded at memory location X'8000'.

or, by typing,

```
#LOAD file [@hex location]
```

```
#START [@hex location] [ program argument]
```

This format is generally used when special loading and execution procedures must be followed. The default device is USR:. The default extension is .OBJ.

Example

#LOAD SYS:RAID

**Bias 6FD0
Start 6FD0
End 90DA**

#LOAD TEST @90DA

**Bias 90DA
Start 90DA
End 9154**

#STA @6FD0

Before the program RAID can be executed, two programs must be loaded into memory: the system program SYS:RAID.OBJ and the user program USR:TEST.OBJ. The first LOAD command loads the system program SYS:RAID.OBJ. The second LOAD command loads the user program USR:TEST.OBJ at the next available memory location (X'90DA'). The START command begins execution of the system program RAID by specifying the starting address in memory of the program (X'6FD0').

An alternate method is to first load the user program, then load and execute the system program RAID by specifying the program name and the next available memory location:

#LOAD TEST

**Bias 6FD0
Start 6FD0
End 714A**

#RAID @714A

Concise Command Language (CCL)

Concise Command Language (CCL) enables the operator to type the program name and the program argument on one line of keyboard input when running a program. The program to be run, however, must make use of the CCL routine in the GOS command processor. (For CCL programming information, see the Plot 80: GOS Programmer's Reference manual.) The operator can use the CCL format when running any GOS utility program.

The following examples show four ways to delete a file by running the GOS utility program DELETE. The GOS utility program file is SYS:DELETE.OBJ and the argument file (the file to be deleted) is USR:FUDD.OBJ. The first example does **not** use CCL, but the last three examples all make use of CCL. Note that in all the examples, the default device for the argument filename is USR:.

Example

Not using CCL:

type,

```
#DELETE
$FUDD.OBJ
```

The DELETE program first displays the character \$ on the screen to prompt the operator for the required argument (the file to be deleted). Using CCL, however, enables the operator to type the argument on the same line as the program name when running a program.

Using CCL:

type,

```
#DELETE FUDD.OBJ
```

or, type,

```
#RUN SYS:DELETE FUDD.OBJ
```

or, type,

```
#LOAD SYS:DELETE
#START FUDD.OBJ
```

Wild Cards

Wild cards are used in a file specifier to represent any or all characters in the filename or extension fields. Wild cards cannot be used in the device field. There are two wild card characters: *(asterisk) and ? (question mark):

*

The wild card * represents any set of characters in the filename or extension field.

Example

*.OBJ

Specifies all files with .OBJ extensions in the specified file structure's directory.

FUDD.*

Specifies all files with filename FUDD in the specified file structure's directory.

.

Specifies all files in the specified file structure's directory.

?

The wild card ? represents any character in a specified character position in the filename or extension field.

Example

FUDD??OBJ

Specifies all .OBJ files in the specified file structure's directory with FUDD as the first 4 characters of the filename and any characters (or no characters) in the remaining 2 positions.

The wild card character *(asterisk), when specified at the end of a filename field, can substitute for a series of wild card ?'s (question marks). For example, typing FUD*.OBJ is equivalent to typing FUD???.OBJ.

For the GOS utility programs, the only wild card allowed in a filename argument to the left of the "=" sign is the *. The *, in this case, represents the corresponding filename field specified to the right of the "=" sign.

Example

```
#COPY CT1:*.*=FD1:FUDD.OBJ,FD1:ELMER.OBJ
```

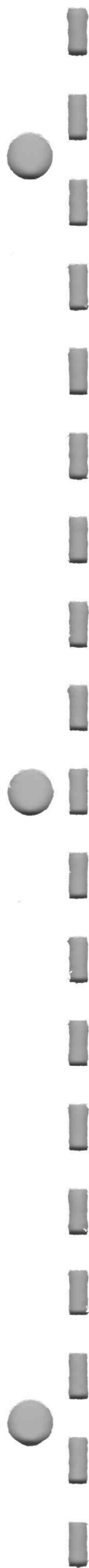
Copy the files FD1:FUDD.OBJ and FD1:ELMER.OBJ to the tape cartridge inserted in the second Cartridge Tape Unit drive. The new files are called CT1:FUDD.OBJ and CT1:ELMER.OBJ.

When specifying a nested file (a file that is a member of a library file), wild cards are allowed only in the last file specified.

Example

```
#DELETE LIB1/LIB2/*.BAK
```

Delete all files with an extension of .BAK in the library USR:LIB1/LIB2.LIB.



SECTION 11

GOS UTILITY PROGRAMS

CONTENTS

	Page
Switches.....	11-6
The Programs	11-8
ATTRIB	11-8
BATCH.....	11-10
COPY.....	11-14
DELETE	11-19
DIR	11-21
DISPLA.....	11-25
FORMAT	11-27
HELP.....	11-33
PDBDMP	11-35
PLOT	11-37
PRINT.....	11-39
RENAME	11-40
SET.....	11-42
SQUISH.....	11-49
SYSTAT.....	11-51
TYPE.....	11-55

Section 11

GOS UTILITY PROGRAMS

The GOS utility programs are stored in files on the system disc or tape. They are available to facilitate many of the routine tasks of a 4081 operator. The GOS utility programs include:

- ATTRIB
- BATCH
- COPY
- DELETE
- DIR
- DISPLA
- FORMAT
- HELP
- PDBDMP
- PLOT
- PRINT
- RENAME
- SET
- SQUISH
- SYSTAT
- TYPE

The 4081 must be operating in Command mode before a utility program can be run. Generally, when the 4081 is operating in Command mode, the prompt character **#** is displayed on the screen. To make sure that the 4081 is operating in Command mode, press the SHIFT-ESC keys. The message **<Attn>** should appear on the screen followed by the prompt character **#**.

Also, before running a utility program, make sure that the system disc or tape (containing the utility programs) is inserted in the file-structured device drive that is assigned the device mnemonic SYS:. To find out which device drive is assigned the SYS: mnemonic, see the utility program **SYSTAT** in this section.

To run a GOS utility program using the Concise Command Language (CCL) format, type the name of the program and the required arguments, then press the RETURN key or the LF key. The program name and arguments can be typed using either upper or lower case characters. After the program is completed it automatically terminates and returns operation of the 4081 to Command mode.

When running a utility program without using the CCL format (in other words, typing just the program name without any arguments), a \$ (dollar sign) or * (asterisk) is displayed on the screen to prompt the operator for the required arguments. After an argument is typed, followed by pressing the RETURN key or LF key, most of the GOS utility programs do not terminate automatically. Instead, the program continues to prompt the operator for additional arguments by displaying another \$ or * prompt character. In this case, to terminate the utility program and return operation of the 4081 to Command mode, either type an "!" (exclamation mark) and press the RETURN key (or LF key) or press the SHIFT-ESC keys.

Example

```
#SET  
$TIME 12:34:00  
$DATE 18-JUN-77  
$!  
#
```

The SET utility requires an argument. Since no argument was specified, a \$ is displayed. The argument TIME 12:34:00 is typed to set the time. Another \$ is displayed and the argument DATA 18-JUN-77 is typed to set the date. Another \$ is displayed and an "!" is typed to exit from the SET utility program.

Table 11-1 explains most of the phrases and symbols used in the examples of GOS utility program syntax in this section.

Table 11-1
GOS UTILITY PROGRAM SYNTAX SYMBOLS

file	A GOS standard file specifier. The GOS standard file specifier has the following format: device:filename.extension
device	A device mnemonic. The GOS device mnemonic consists of 2-3 characters followed by a colon(:).
program argument	A value, device, or file that is operated on by a program.
[]	Information enclosed in brackets is optional.
/	Library file identifier (e.g. the file SYSEQU.ASM in the library SYSLIB.LIB is specified as SYSLIB/SYSEQU.ASM).
:	1. Device identifier; follows the device mnemonic (e.g. FDO:). 2. Separates a switch from its value (e.g. ;N:NAME).
;	Switch identifier; precedes the switch (e.g. ;H).
.	Filename extension identifier; precedes the extension (e.g. .OBJ).
#	GOS command processor input request.
*, \$	General purpose input requests (input depends on the program running).

NOTE

In the examples for the utility programs, the characters printed in boldface type represent the information that is actually displayed on the screen by GOS or the running program. The operator need type only the characters that are not printed in boldface type.

Switches

A switch specifies a program option. When running a GOS utility program, switches can be typed on the same line of keyboard input as the utility program name. A switch is specified as a letter preceded by a semi-colon (;).

Example

```
#DIR;B
```

Display the directory of the USR: device and, since the ;B switch was specified, include a listing of any bad blocks in the directory. If the ;B switch is not specified, bad blocks are not included in the directory listing.

Most switches specify program options that are unique to a particular utility. The function of each valid switch is detailed in the utility program description. The ;H and ;V switches are valid options for many of the GOS utility programs. The general functions of these two switches are explained here:

;H Display a brief help message for the specified GOS utility program. A \$ or * is displayed after the message to prompt the operator for a program argument. The ;H switch is typed directly following the GOS utility program name or after the utility prompts the operator for a program argument. An ;H switch help message is available for every GOS utility program except HELP, SYSTAT, PDBDMP, PLOT, and PRINT.

Example

```
#DELETE ,H
```

```
DELETE Version 3 0 Level 0
FNAME{ ,FNAME, ,FNAME}{ ,V}
,V - To allow user verification of files to be deleted
Y - Signifies file should be deleted
$
```

Display a help message for the DELETE utility. The prompt character \$ is then displayed to indicate that an argument (the file to be deleted) is required.

;V Verify each file to be used as a GOS utility program argument. The filename is displayed on the screen followed by a ? to prompt the operator for a response. If a "Y" is typed in the first character position (e.g., "YES"), the filename becomes a

verified argument and is transferred to the utility program. If any character other than a "Y" is typed in the first character position, the filename is ignored. All responses must be followed by pressing the RETURN key. The ;V switch is especially useful when wild card filenames are specified as program arguments.

Example

```
#DELETE *.BAK;V
USR:ELMER.BAK ?
USR:TEMP.BAK ?Y
USR:BAKER.BAK ?
USR:LOSER.BAK ?Y
      14 Blocks released
#
```

All .BAK files on the user device are displayed, one filename at a time. If a "Y" typed after the filename, that file is deleted. If only the RETURN key is pressed (or any character other than a "Y" is typed), the file is not deleted. In this example, only USR:TEMP.BAK and USR:LOSER.BAK are deleted.

The operator can specify all switches except the ;H switch in any order. The utility program syntax shows the proper placement of all switches. Generally, the error message **Illegal switch** is displayed if an invalid switch is specified.

THE PROGRAMS

ATTRIB

Syntax

ATTRIB[;H] file[;W] [;V] [,file ...]

Defaults

for a file, device: =USR:
 .extension = none

Wild Cards

Wild cards are allowed.

Switches

- ;H Display the ATTRIB utility help message.
- ;W Write-protect attribute: file cannot be deleted, appended, or the contents changed in any way.
- ;R Read-protect attribute: file cannot be read (in other words, it cannot be executed, copied, deleted, displayed, etc.).
- ;V Verify each file before assigning attributes.

If no switches are specified, all of the file's attributes are removed.

Description

Assign read and write-protect attributes to a file by specifying the ;R and ;W switches. If a write and/or read-protected file is attributed again without specifying a ;W or ;R switch, then the file's read and/or write-protect attributes are removed.

To assign or remove attributes, the directory of the file structure that contains the file must be changed. The directory cannot be changed if the medium or device drive is write-protected. If an attempt is made to attribute a file on a write-protected medium or device drive, the error message **Dir I/O error IF lu 1** is displayed.

Example

```
#ATTRIB *.OBJ;W
```

Write protect all .OBJ files on the USR: device. If any of the .OBJ files were read-protected, the read-protect attribute is automatically removed.

```
#ATTRIB CT1:*.OBJ
```

Remove read/write-protect attributes from all .OBJ files on the device CT1:.

```
#COPY WHITE OBJ-FD1 BLACK OBJ
-Files Copied-
File protect error
None
#DIR FD1 BLACK OBJ
```

```
BLACK OBJ [12] 23-May-77
#ATTRIB FD1 BLACK OBJ,W
FD1 BLACK OBJ
#COPY WRITE OBJ-FD1:BLACK OBJ
-Files Copied-
FD1 BLACK OBJ
#ATTRIB FD1 BLACK OBJ,W,R
FD1 BLACK OBJ
```

The first COPY command attempts to copy FD1:BLACK.OBJ to USR:WHITE.OBJ, but the error message **File protect error** is displayed. The DIR utility is run to display the directory entry for the file FD1:BLACK.OBJ. The directory entry shows that the file is both read and write-protected. The read-protection attribute prevents the file from being copied. The ATTRIB utility is then run to remove the read-protect attribute but retain the write-protect attribute on the file FD1:BLACK.OBJ. The COPY procedure is again attempted and this time the file is copied successfully. The ATTRIB utility is run again to read and write-protect the file FD1:BLACK.OBJ.

BATCH

Syntax

BATCH[;H] file[;L]

Defaults

for a file, device:=USR:
 .extension=.BAT

Wild Cards

Wild cards are *not* allowed.

Switches

;H Display the BATCH utility help message.

;L List each command in the batch file as it is executed.

Description

Run a sequence of commands listed in the specified batch file. The command sequence can include any combination of GOS resident commands, GOS utility programs, and user programs.

NOTE

Under GOS, batch files are created using the program file MAKE.OBJ (resident on the system disc or tape). The procedure described here is valid only on a 4081 equipped with the optional 4080A01 Plot 80: Programming Support Package. For a more detailed explanation of the creation and editing of files, see the Plot 80: GOS Text Editor & Corrector (TECO) manual.

A batch file is created using the following procedure:

1. Type "MAKE filename.BAT". The .BAT extension identifies the file as a batch file and allows the operator to execute the file without typing the extension. Press the RETURN key and wait for the *(asterisk) prompt character to be displayed on the screen.

```
#MAKE filename .BAT
#
```

2. Type "I", the instruction for inserting text in a file. All characters typed following the "I" are inserted in the file.

```
#MAKE filename .BAT
*I
```

3. Type the commands to be inserted in the batch file. Press the RETURN key after each command. In case of a typing error, press the RUBOUT key to delete the last character typed on the current line (the deleted character is displayed again); press the ESC key to delete the entire current line (is displayed on the line after the deleted line).

```
#MAKE filename.BAT
*Icommand
command
-
-
-
```

4. After typing all the commands to be inserted in the batch file, press the LF key. A ¶ is displayed on the screen. The LF key separates instructions. Now type "EX", the instruction for exiting the MAKE program, then press the LF key twice. ¶ ¶ is displayed on the screen. Pressing the LF key twice executes all the preceding instructions. The typed commands are inserted in the file, the MAKE program exits (**Exit** is displayed on the screen), and the 4081 enters Command mode (**#** is displayed on the screen).

```
#MAKE filename.BAT
*!command
command
-
-
-
¶ EX ¶ ¶
Exit
#
```

When a GOS utility program is listed in a batch file without specifying a program argument, the program will not exit automatically when the batch file is executed. Instead, the program continues to prompt the operator for arguments. If the operator types a "!" (exclamation mark) at this point and then presses the RETURN key, the utility program exits normally and the next command in the batch file is executed.

After all the commands in the batch file are executed, the following message is displayed:

-> End of Batch Stream

Example

```
#MAKE VAMPYR BAT  
  
*ICHA 5  
ATTRIB FUDD OBJ ;W  
RUN FUDD 08000  
CLO  
EXIT  
#BATCH VAMPYR ;L
```

By typing "BATCH VAMPYR" the batch program USR:VAMPYR.BAT is executed. The batch program sets the Display Monitor's character size to 5, assigns a write-protect attribute to the file USER:FUDD.OBJ, runs the program load file USR:FUDD.OBJ after loading it at memory location X'8000', then executes the GOS resident command CLOSE. By specifying the ;L switch, each command in the batch file is displayed on the screen as it is executed.

COPY

Syntax

COPY[;H] new file or destination device=old file or source device [old file . . .] [;V] [;B]

Defaults

for a file, device:=USR:
 .extension=none

for a device, device=none

Wild Cards

The wild card * can be used in filenames on either side of the "=" sign.

The wild card ? can be used only on the right side of the "=" sign.

Switches

- ;H Display the COPY utility help message.
- ;V Verify each old file or source device before copying.
- ;B Allow binary graphics files to be copied (displayed) on the Plotter (PL:) or Display Monitor (DC:). Binary graphics files usually have the extension .PLT. Using this switch with the COPY utility performs the same procedure as the PLOT and DISPLA utility programs.

Description

Copy the data in the old files to the new file. The old filenames are displayed as each is copied. If an old file does not exist, it cannot be copied and the filename is not displayed. If none of the old files exist, no files are copied and the word **None** is displayed.

The attributes of the old file are also copied to the new file. So, if the old file was assigned a write-protect attribute (the ;W attribute), then the new file is also assigned a write-protect attribute.

NOTE

*A read-protect file cannot be copied. The error message **File protect error** is displayed if an attempt is made to copy a read-protect file.*

*A destination device must not be write-protected. The error message **Dir I/O error IF lu 2** is displayed if an attempt is made to copy a file to a write-protected device.*

*Also, the ;B switch can only be used to copy a binary graphics file. The error message **I/O error UE lu 2** is displayed if the ;B switch is used to copy a file other than a binary graphics file. Picture Data Base files are not binary graphics files. They are stored in a special format.*

Normally, the new file is allocated storage according to the GOS automatic file sizing procedure. The file size (in blocks) can be specified in square brackets after the filename. (See the heading **File Sizing** in the section **Files** for more details.) If the wild card * is used in either the filename or extension fields or both the filename and extension fields of the new file specifier (for example FD1:*.*/), then the GOS automatic file sizing procedure is not used to allocate space for the new file. Instead, the size of the new file is the exact size of the file being copied. If only a file-structured device is specified, the wild card * is assumed for both the filename and extension fields (FD0: is equivalent to FD0:*./*).

Example

```
#COPY FD2:*.*=ROCKET.V2
```

or

```
#COPY FD2:=ROCKET.V2
```

Makes an exact copy of the file `USR:ROCKET.V2` on the flexible disc in `FD2:`. The new file is also named `ROCKET.V2`.

If file-structured devices are specified for both the source and destination, then all the files on the source device are copied, file by file, to the destination device.

Example

```
#COPY FD1:=FD2:
```

or

```
#COPY FD1:*.*=FD2:*./*
```

Copies each file on `FD2:` to `FD1:`. The files on `FD1:` are the same size and have the same names as the files on `FD2:`.

If more than one old file is specified and no wild cards are used to specify the new file, the old files are combined and copied to the new file. If the ;B switch is specified, only binary graphics files can be combined. The error message **I/O error UE lu 2** is displayed if an attempt is made to combine a binary graphics file with any other type of file when the ;B switch is specified.

Example

```
#COPY FD1:FILE.LST=FD2:*ASM
```

Combines all the files with the extension .ASM on the flexible disc inserted in FD2: into one file and copies the file to FD1:FILE.LST.

With library files, either the entire library file structure or the individual files in the library can be copied. If the source file is a library, the entire library structure is copied by specifying the library filename and the extension. The destination file is always a library. The new library does not need to be previously formatted. The structure identifier of the source library is copied to the structure identifier of the destination library.

Example

```
#COPY DK1:MENS.LIB=WOMENS.LIB
```

Copies the entire library structure USR:WOMENS.LIB to create the new library DK1:MENS.LIB.

The contents of a library can be copied by following the library name with / or /*.* (the notations are equivalent). Individual members of a library are copied by specifying the member's filename and extension after /.

Example

```
#COPY CT1:=WOMENS/
```

or

```
#COPY CT1:=WOMENS/*.*
```

Copies each member of the library USR:WOMENS.LIB to a corresponding non-library file with the same size, filename, and extension on the tape cartridge inserted in CT1:.

```
#COPY FD2:=BAKER/BREAD.ASM,BAKER/DOUGH.NUT
```

Copies the files BREAD.ASM and DOUGH.NUT in the library USR:BAKER.LIB to corresponding non-library files with the same size, filename, and extension on the flexible disc inserted in FD2:.

If a library is specified as the destination and non-library files are specified as source files, the library must be previously formatted and the library identifier / must be used to retain the library structure. If the / is not used, a non-library file is created, whether or not the library was previously formatted. If the library was not previously formatted and the / is used, the error message **"file" not found** is displayed.

Example

#COPY CITY.LIB=ELMER.OBJ

Copies the file USR:ELMER.OBJ to the non-library file USR:CITY.LIB.

#COPY CITY/ELMER.VIL=ELMER.OBJ

Copies the file USR:ELMER.OBJ to create the new file ELMER.VIL in the library USR:CITY.LIB. The library USR:CITY.LIB was previously formatted.

#COPY WOMENS/=CT1:

Copies all the files on CT1: to create corresponding new files in the library USR:WOMENS.LIB. The new library members have the same attributes, size, filename, and extension as the source files on CT1: The library USR:WOMENS.LIB was previously formatted.

NOTE

Following the filename with / defaults the library filename extension to .LIB. If the standard library extension .LIB is not used, precede the / with the library extension used (for example, PACKING.BOX/.* specifies all members of the library PACKING.BOX).*

Example

#COPY DK0:=CT0;;V

Copy all files on CT0: to DK0:. Before each file is copied, the source file's name is displayed followed by ?. By typing "Y", the file is copied. If "Y" is not the first character typed, the file is not copied. The new files have the same attributes, size, filename, and extension as the source files. This form of the COPY program can be used to copy the system utility tape cartridge to a formatted disc (in this case, the hard disc DK0:).

#COPY MYLIB.LIB[100]=FD1:YOUR.LIB

Copy the entire library FD1:YOUR.LIB to create the new library USR:MYLIB.LIB. The new library is allocated 100 blocks since the size was specified (see **File Sizing** in the section **Files**).

#COPY DC:=SYS:DEMO/LANDER.PLT;B

Display the binary graphics file LANDER.PLT in the system library DEMO.LIB. The ;B switch must be specified or the file cannot be displayed. This form of the COPY program is equivalent to typing:

#DISPLA SYS:DEMO/LANDER.PLT

DELETE

Syntax

DELETE[;H] file [,file . . .] [;V]

Defaults

for a file, device:=USR:
 .extension=none

Wild Cards

Wild cards are allowed.

Switches

;H Display the DELETE utility help message.

;V Verify each file before deleting.

Description

Delete a file from a file structure. The space previously occupied by the file is available for the storage of new files.

To delete a file, the directory entry for that file must be changed. The directory entry cannot be changed if the medium is write-protected. If an attempt is made to delete a file on a write-protected medium, the error message **Dir I/O error IF lu 1** is displayed.

A file that has read or write-protect attributes cannot be deleted. If the file is read-protected, the error message **Locked file** is displayed. If the file is write-protected or both read and write-protected, the error message **File protect error** is displayed.

With libraries, either the entire library or individual files in the library can be deleted. To delete an entire library, the filename and extension must be specified. To delete the file in the library, the library name must be followed by / and the member's filenames and extensions (or wild cards).

Example

#DELETE WOMENS.LIB

Deletes the entire library structure USR:WOMENS.LIB.

#DELETE WOMENS/*.*

Deletes all the files in the library USR:WOMENS.LIB.

NOTE

Following the filename with / defaults the library file extension to .LIB. If the standard library extension .LIB is not used, precede the / with the library extension used (e.g., PACKING.BOX/.* specifies all members of the library PACKING.BOX).*

Example

#DELETE FUDD.OBJ

Delete the file USR:FUDD.OBJ.

#DELETE ELMER.*;V

USR:ELMER.BAK ?Y

USR:ELMER.ASM ?Y

USR:ELMER.OBJ ?

USR:ELMER.LST ?

28 Blocks released

#

All files named ELMER on the user device are displayed, one filename at a time. If a "Y" is typed after the file's name, the file is deleted. If only the RETURN key is pressed (or any character other than a "Y" is typed), the file is not deleted. In this example, only USR:ELMER.BAK and USR:ELMER.ASM are deleted.

#DELETE ZEPLIN.LED/PLANT.BOB

Delete the file PLANT.BOB from the library USR:ZEPLIN.LED.

#DELETE MENS/BOB.PDB,MENS/HARRY.PDB

Delete the members BOB.PDB and HARRY.PDB from the library USR:MENS.LIB.

DIR

Syntax

DIR[;H] device or file [;B] [;E] [;F] [;F:n] [;L] [destination device or file]

Defaults

for a device, device:USR:

for a file, device:=USR:
 .extension=.* (any extension)

for a destination device, device:=DC:

for a destination file, device:=USR:
 .extension=.LST

Wild Cards

Wild cards are allowed.

Switches

- ;H Display the DIR utility help message.

- ;B Include listing of bad blocks in directory; bad blocks are listed as **BLKxxx.BAD** (xxx represents the block number of the bad block on the device in hexadecimal notation). Bad blocks are automatically write and read-protected.

- ;E Include listing of empty blocks (free space) in directory. Empty blocks are listed as **<Empty>**.

- ;F Fast format directory listing: only the name and size of each file is listed and the directory is displayed in four columns across the screen.

- ;F:n Fast format directory listing: the integer value **n** specifies the number of characters to be displayed per line on the screen. (A column of directory entries requires 20 characters; therefore, the number of columns equals n/20).

- ;L Do not list individual files in the directory.

Description

Display a directory of the files on the specified file structure. The directory information is displayed in the following format:

Directory Structure of (device:structure identifier) (date structure formatted)

Directory Blocks: Available=(number available) Used=(number used)

(filename)	(size of file in blocks)	(date and time of creation)	(attributes)
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-

Files=(number of files), Blocks=number of blocks currently in use)
Free=(number of blocks of empty space) Largest=(largest number of adjacent blocks of empty space)

NOTE

If the time is not set when the device is formatted or the file created, the directory entry for time is blank.

An L attribute indicates a library file.

If a non-file-structured file (a file other than a library file) is specified, the DIR utility displays the file's name, size, date and time created, and any attributes as listed in the file structure directory entry.

If a file is specified that is not listed in the specified file structure's directory, then the message **--None Listed--** is displayed on the screen.

If a destination device or file is specified, the directory information is copied to the device or file (in this case, the directory information is not displayed). The file size (in blocks) can be specified after the destination file name. If the file size is not specified, the GOS automatic file sizing procedure is used (see **File Sizing** in the section **Files** for more details).

Example

```
#DIR FD1 ,B,E
```

Directory Structure of FD1:MYDISC 23-Jun-77
Directory Blocks: Available- 8, Used- 1

LINES	PDB	[1]	27-Sep-76		
<Empty>		[1]			
EXAMPL	ASM	[1]	17-Jan-77		W R
EXAMPL	OBJ	[1]	17-Jan-77		W
NEW	OBJ	[30]	1-Jan-77	11.33	
LABOUT	OBJ	[23]	21-Sep-76		
NEW	FOR	[30]	23-Jun-77	13.00	W
<Empty>		[257]			
BLK160	BAD	[16]	23-Jun-77	12.53	W R
MASTER	LIB	[400]	23-Jun-77	12.56	L
<Empty>		[464]			

Files-8,Blocks-502,Free-722,Largest-464

Display a listing of the files on the flexible disc in FD1: and include a listing of the bad blocks and empty blocks on the disc.

```
#DIR ,F 60
```

Directory Structure of USR:MYDISC 23-Jun-77
Directory Blocks: Available- 8, Used- 1

LINES	PDB	[1]	EXAMPL	ASM	[1]	EXAMPL	OBJ	[1]
NEW	OBJ	[30]	LABOUT	OBJ	[23]	NEW	FOR	[30]
MASTER	LIB	[400]						

Files-8,Blocks-502,Free-722,Largest-464

Display a listing of the USR: device directory using 60 characters per line (3 columns) on the screen.

#DIR EXAMPL

EXAMPL ASM	[1]	17-Jan-77	W R
EXAMPL OBJ	[1]	17-Jan-77	W

Display a listing of all files on the USR: device with the filename EXAMPL and any extension.

#DIR FD2:PUBLIC.LIB;L

Display just the directory entry for the library FD2:PUBLIC.LIB. Do not include individual files within the library in the listing.

#DIR DK1:ELMER.LIB,LP:

Copy a listing of the directory of the library DK1:ELMER.LIB to the Line Printer. The listing is not displayed on the screen.

#DIR FD:BODY/LEG.LIB,DIRLEG

Copy a listing of the directory of the library LEG.LIB in the library USR:BODY.LIB to the file USR:DIRLEG.LST. The listing is not displayed on the screen.

DISPLA

Syntax

DISPLA[;H] graphics file[,graphics file ...]

Defaults

```
for a graphics file,      device:=USR:
                           .extension=.PLT
```

Wild Cards

Wild cards are allowed.

Switches

```
;H    Display the DISPLA utility help message.
```

Description

Display the contents of a graphics file on the screen. A graphics file contains graphics data in the form of screen X, Y coordinates. Generally, graphics files are created by storing the output of SVC 15 or FORTRAN graphics instructions in a file.

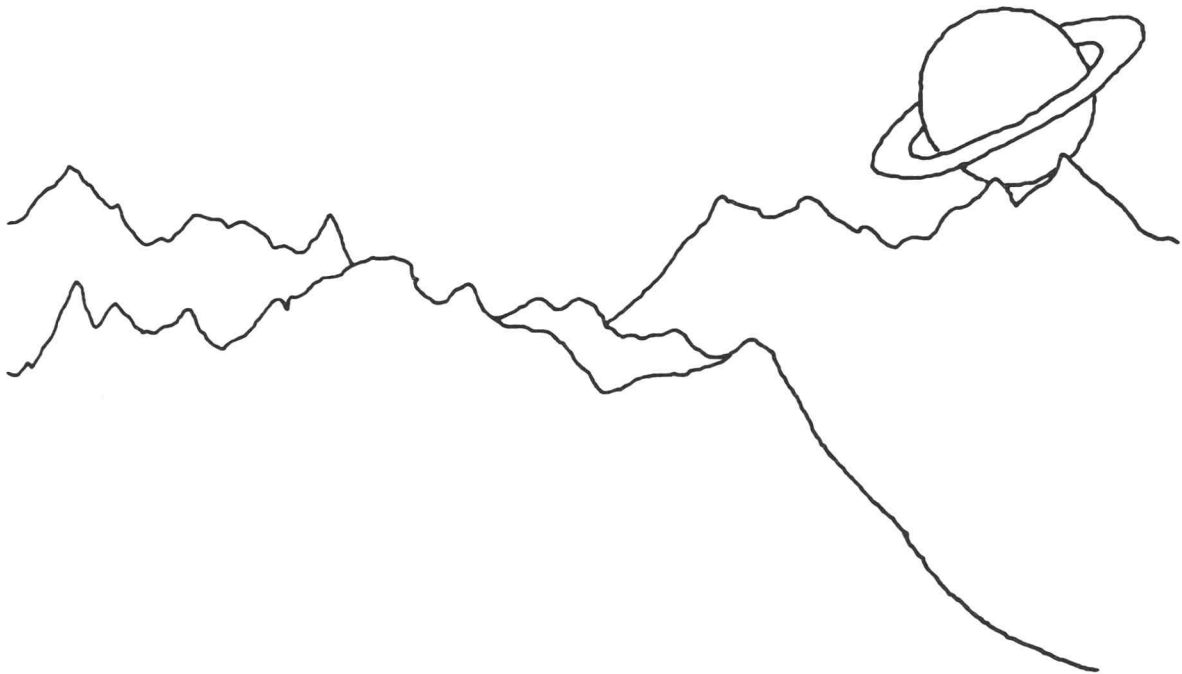
If two or more graphics files are specified, the files are combined and then displayed.

If a non-graphics file is specified, the error message **I/O error UE lu 2** is displayed. Picture Data Base files *cannot* be displayed by the DISPLA utility. They are stored in a special format.

Press the CTRL-S keys to stop the display of the contents of a graphics file on the screen. Press the CTRL-S keys again to continue the display. Press the CTRL-O keys to prevent the display of the remaining contents of a graphics file on the screen. (See **CTRL-O** and **CTRL-S** in the section **Switches, Messages, Modes, Lights, and Keys** for more details.)

Example

```
#DISPLA SYS:DEMO/LANDER
```



Display the graphics file LANDER.PLT. on the Display Monitor. LANDER.PLT is in the library SYS:DEMO.LIB.

```
#DISPLA DOUBLE,FEATUR
```

Combine the graphics files USR:DOUBLE.PLT and USR:FEATURE.PLT, then display the resulting picture.

FORMAT

Syntax

FORMAT[;H] device or file[[library size]] [;N:name] [;E:n] [:X:n]

Defaults

for a device,	device:=none
for a file,	device:=USR: .extension=.LIB

Wild Cards

Wild cards are *not* allowed.

Switches

;H Display the FORMAT utility help message.

;N:name **name** is the structure identifier. It is 1-6 characters long. The structure identifier can be used by the operator to further identify the file structure (it is ignored by GOS and is not part of the GOS standard file specifier). The structure identifier appears in the directory listing (see the DIR utility). When formatting a device, the ;N switch is required. When formatting a library file, the ;N switch is optional (if this switch is not specified, the library filename is the assumed structure identifier). Only the following characters can be used to name structure identifiers:

ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789#\$\$%

;E:n **n** is the number of directory entries to be reserved for the file structure. Table 11-2 contains the default value of **n** for the specified device or file (if the ;E switch is omitted):

Table 11-2
DEFAULT NUMBER OF DIRECTORY ENTRIES

File Structure	Number of Directory Entries (default)
DKn:	720
FDn:	120
CTn:	120
Library File	60

;X:n **n** is the number of extra bytes to be reserved for each directory entry in the file structure. **n** must be an even number. The default value of **n** is 0 (if the ;X switch is omitted).

Description

Format a file-structured device (actually the medium inserted in the device) or library file. The FORMAT utility prepares a medium or library for the storage of files. A formatted medium or library file is called a file structure. Formatting a medium or library file destroys all previous data stored in the file structure.

NOTE

The date should always be set before formatting a medium or library file. If the date is not set, the directory entry for date will be a default value (generally, the date that the GOS IPL tape was created). If library files are being formatted, the time should also be set before formatting.

If a device is specified, for example,

#FORMAT FD1:

the following message is displayed:

Mount volume to be formatted
Type a "G" to Go.

The operator then inserts the medium to be formatted, types "G", presses the RETURN key, and this message is displayed:

Formatting in progress

When the formatting is completed, the message **Formatting done** is displayed. A message also informs the operator of the number of bad blocks, if any, on the medium. If any other character is typed before "G", the message **Improper response, program terminating** is displayed and operation of the 4081 is returned to Command mode.

If a filename is specified, for example,

#FORMAT PUBLIC.LIB[80]

a library is formatted. If the library has not been previously formatted, the size of the library in blocks must be specified after the filename (the library size must be enclosed in square brackets, for example, [80]). If the library has been previously formatted and the size is not specified, a new library of the same size and name is created. When formatted, the library is automatically assigned an L attribute (to identify the file as a library) and the message **Formatting done** is displayed.

NOTE

Formatting a tape cartridge (CT:) requires a lengthy amount of machine time. The 4081 cannot do any other processing while formatting is in progress. Also, formatting a tape disables clock interrupts; therefore, if the time of day was set previous to formatting a tape, the new time should be set after formatting is completed.

Errors may occur when formatting a medium or library. Table 11-3 lists some common error messages and some probable causes for the messages.

Table 11-3
FORMATTING ERRORS

Error Message	Description
Dir I/O error DU lu 2	This message occurs when an attempt is made to format a library and the device is unavailable. A device is unavailable if the medium is not inserted in the device, the power is not ON, or the drive door is open.
Dir I/O error IF lu 2	This message occurs if an attempt is made to format a library and the device or medium is write-protected.
Dir I/O error UE lu 2	If an attempt is made to format a library on a medium that is not formatted, this message is displayed. The medium must be formatted before a library can be formatted.
Fatal error - medium unusable!	If the medium is removed while formatting is in progress, this message is displayed. The medium must be reformatted. This message is also displayed when a bad block is detected in the area of the medium on which the primary directory is to be created. In this case, try formatting the medium again. If the same error message is displayed, use a new medium.
"File" not found	When formatting a new library, this error message is displayed if the library size was omitted. The library size is optional only when the library exists and the operator desires to reformat the library.

Error Message	Description
Medium is write protected Pause	This message occurs after the message Formatting in progress if an attempt is made to format a write-protected medium. Either the medium itself or the device is write-protected. After removing write protection, the operator can continue formatting by executing the resident command CONTINUE.
More dir entries than space available!	This message occurs if the size of a library file is not large enough to contain the required number of directory blocks (generally, when the file is less than or equal to four blocks in size). Specify a larger number of blocks for the library file's size or reduce the number of directory entries specified (using the ;E switch) when running the FORMAT program again.
Please mount disk or Please mount tape cartridge for write pass	These messages occur after the message Formatting in progress if an attempt is made to format a medium when the device is unavailable. A device is unavailable if the medium is not inserted in the device, the power is not ON, or the drive door is open. After correcting the problem, the operator can continue formatting by executing the resident command CONTINUE.
Str full!	When formatting a library, if the specified library size is greater than the size of the largest number of adjacent empty blocks, this message is displayed. (See SQUISH in this section for information on possible solutions.)

Error Message	Description
Volume I.D. required! *	If the structure identifier is not specified when formatting a device, this message appears after Formatting in progress and the number of bad blocks is displayed. The operator must enter the structure identifier in the same format required in the command string (;N:name). Additional asterisks are displayed until the structure identifier is correctly entered. The message Formatting done is then displayed.

Example

#FORMAT DK1;;N:BOB

Format the device DK1: and assign it the structure identifier **BOB**. The directory will contain space for 720 filename entries (the default value for the hard disc).

#FORMAT USR;;N:MAYBE;E:150;X:10

Format the user device and assign it the structure identifier **MAYBE**. The directory will contain space for 150 filename entries and each directory entry will contain 10 extra bytes.

#FORMAT DK1:BUGS[80]

Format the library file DK1:BUGS.LIB and assign it the structure identifier **BUGS**. The directory will contain space for 60 entries (the default value for library files). The library file will be 80 blocks in size.

#FORMAT MARLOW[100];N:PHIL;E:50

Format the library file USR:MARLOW.LIB and assign it the structure identifier **PHIL**. The directory will contain space for 50 entries. The library file will be 100 blocks in size.

HELP

Syntax

HELP utility program name [,destination device or file]

Defaults

for a utility program name,	device: = none can be specified (assumes SYS:) .extension = .HLP
for a destination device,	device: = DC:
for a destination file,	device: = USR: .extension = .LST

Wild Cards

Wild cards are *not* allowed.

Switches

none

Description

Display a help message for the specified GOS utility program. A help message contains information on the purpose, syntax, and sample uses of the utility program.

Help messages are available only for the utility programs listed in SYS:HELP.LIB (a library of help message files on the system disc or tape). Type HELP to get a listing of the available help messages. After the listing is displayed a \$ appears to prompt the operator. Any of the utility program names listed can be entered when the \$ appears. The help file message for that utility is then displayed and another \$ appears to request another utility name. To exit from the HELP program either type "!" or press the SHIFT-ESC keys.

NOTE

*The system disc or tape that contains the **HELP** utility program must be inserted in the device drive assigned the device mnemonic **SYS:**. The error message **HELP.LIB is not on the present system disc** (or tape) is displayed if **SYS:** is assigned to another device.*

If a destination device or file is specified, the help message is copied to that device or file (in this case, the message is not displayed). The destination file size (in blocks) can be specified after the destination filename. If the file size is not specified, the GOS automatic file sizing procedure is used (see **File Sizing** in the section **Files** for more details).

Generally, the help message displayed using the **HELP** utility is more detailed than the help message displayed using the ;H switch. **HELP** messages are available for **HELP**, **SYSTAT**, and **PDBDMP** when the **HELP** utility is used. A help message detailing GOS standard error messages is also available.

Example

#HELP FORMAT,LP:

Copy the help message for the **FORMAT** utility to the Line Printer. The help message is not displayed on the screen.

#HELP ERROR,ERR

Copy the help message describing GOS standard error messages to the file **USR:ERR.LST**. The help message is not displayed on the screen.

#HELP HELP

Display the help message for the **HELP** utility on the screen.

#HELP

Display a listing of all the available help messages. After the listing is displayed, the **\$** prompt character appears. Any of the GOS utility program names listed can be entered.

PDBDMP

Syntax

PDBDMP [destination device or file=]Picture Data Base file

Defaults

for a destination device,	device: = DC:
for a destination file,	device: = USR: .extension = LST
for a Picture Data Base file,	device: = USR: .extension = .PDB

Wild Cards

Wild cards are *not* allowed.

Switches

none

Description

Display an ASCII listing of the contents of a Picture Data Base file. If a destination device or file is specified, the listing is copied to that device or file (the listing is not displayed). The file size (in blocks) can be specified after the destination filename. If the file size is not specified, the GOS automatic file sizing procedure is used (see **File Sizing** in the section **Files** for more details).

Picture Data Base is a format for the storage of binary graphics information. Picture Data Base files are created by the Plot 80 software packages Interactive Graphics Terminal (IGT) and Digitizer.

Example

```
#PDBDMP STRSHP=STRSHP
```

Copy an ASCII listing of the contents of the Picture Data Base file USR:STRSHP.PDB to the file USR:STRSHP.LST.

```
#PDBDMP BOX1
```

Picture Data Base Dump V3.0L0, File = BOX1

```
Header FDM, Len=22
```

```
Status (Hex)-0000
```

```
X Setpoint-0, Y Setpoint-0
```

```
X Scale (Hex)-0100, Y Scale (Hex)-0100
```

```
Rotation (Hex)-0000
```

```
Extent--100,-109,1520,1506
```

```
Long, Relative, Draw, Len=18
```

```
X=1420, Y=5
```

```
X=-4, Y=1401
```

```
X=-1412, Y=-5
```

```
X=-4, Y=-1410
```

```
#
```

Display an ASCII listing of the contents of the Picture Data Base file USR:BOX1.PDB on the screen.

```
#PDBDMP FD1:NEWD,LP:
```

Copy an ASCII listing of the contents of the Picture Data Base file FD1:NEWD.PDB to the Line Printer.

PLOT

Syntax

PLOT graphics file

Defaults

for a graphics file, device:=USR:
 .extension=.PLT

Wild Cards

Wild cards are *not* allowed.

Switches

none

Description

Draw the contents of a graphics file on the Plotter. The file must contain graphics data in the form of screen X, Y coordinates. Generally, graphics files are created by storing the output of SVC 15 or FORTRAN graphics instructions in a file. Picture Data Base files *cannot* be plotted by the PLOT utility. They are stored in a special format.

The PLOT utility is similar to the DISPLA utility, except the graphics file is drawn on the Plotter, not displayed on the screen.

Other programs can be run in the background while the PLOT utility is running only if the other programs do not use logical units 14 and 15. No other GOS utility program, except PRINT, uses logical units 14 and 15. A background program is processed automatically, under lower priority, when the higher priority foreground program (in this case, the PLOT utility) is not using the 4081 processor.

NOTE

The PLOT utility program can be run only on a 4081 with 64K bytes of memory.

Example

#PLOT EIGHTY

Draw the contents of the graphics file USR:EIGHTY.PLT on the Plotter.

#PLOT DK1:ELMER.FUD

Draw the contents of the graphics file DK1:ELMER.FUD on the Plotter.

PRINT

Syntax

PRINT ASCII file

Defaults

for an ASCII file, device: =USR:
 .extension=.LST

Wild Cards

Wild cards are *not* allowed.

Switches

none

Description

List the contents of an ASCII file on the Line Printer.

The PRINT utility is similar to the TYPE utility, except the ASCII file is listed on the Line Printer, not on the screen.

Other programs can be run in the background while the PRINT utility is running only if the other programs do not use logical units 14 and 15. No other GOS utility program, except PLOT, uses logical units 14 and 15. A background program is processed automatically, under lower priority, when the higher priority foreground program (in this case, the PRINT utility) is not using the 4081 processor.

NOTE

The PRINT utility program can be run only on a 4081 with 64K bytes of memory.

Example

#PRINT DK0:GOS

List the contents of the ASCII file DK0:GOS.LST on the Line Printer.

#PRINT MACH0.ASM

List the contents of the ASCII file USR:MACH0.ASM on the Line Printer.

RENAME

Syntax

RENAME[;H] new file or name = old file or device[;V]
[new file or name = old file or device . . .]

Defaults

for a new file, device: =USR:
 .extension = none

for a new name, name = none

for an old file, device: =USR:
 .extension = none

for an old device, device: = none

Wild Cards

The wild card * can be used in filenames on either side of the "=" sign.

The wild card ? can be used only on the right side of the "=" sign.

Switches

;H Display the RENAME utility help message.

;V Verify each old file or device before renaming.

Description

Rename a filename or structure identifier. The new filename retains the attributes of the old filename; so, if a write-protected (;W attribute) file is renamed, then the new filename is also write-protected.

To rename a filename or structure identifier, the directory entry for that file or file structure must be changed. The directory entry cannot be changed if the medium is write-protected. If an attempt is made to rename a file or file structure on a write-protected medium, the error message **Dir I/O error IF lu 1** is displayed.

If the old file resides on a device other than **USR:** (the default device name) or if it is a member of a library file, the device or library file need only be specified as part of the old filename (on the right side of the "=" sign). The same device and library is assumed for the new filename. The device must be the same for the new and old filenames or the error message **Bad dir/parm!** is displayed.

When renaming a library filename, the library's structure identifier is also renamed (with the same name as the new library filename) unless the library filename is currently write-protected (using the **ATTRIB** utility). If the library is write-protected, only the filename is renamed. The structure identifier is not changed and the message **File locked** is displayed. A library's structure identifier, however, can only be renamed by renaming the library filename.

Example

#RENAME BOB=FD1:

Change the name of the structure identifier for the file structure **FD1:** TO **BOB**.

#RENAME BOB.*=FD1:CHAPPD.LIB

Change the name of the library **FD1:CHAPPD.LIB** to **FD1:BOB.LIB**. If **FD1:CHAPPD.LIB** is not write-protected, then also change the name of the library's structure identifier to **BOB**.

#RENAME BOB.*=FD1:CHAPPD/SAM.ASM

Change the name of the file **SAM.ASM** in the library file **FD1:CHAPPD.LIB** to **BOB.ASM**. The full filename now becomes **FD1:CHAPPD/BOB.ASM**.

SET

Syntax

SET option[,option . . .]

Defaults

none

Wild Cards

Wild cards are *not* allowed.

Switches

none

Description

Assign a mnemonic to a device, set time or date, or set environmental parameters such as MVP boundaries or cursor size. General purpose SET utility options are described in groups in the following text. The SET utility options that are used to initialize data communications are described in the section **Data Communications in Host Mode**

NOTE

If a SET option or its arguments is longer than 3 characters, the operator need type only the first 3 letters.

Option--Assign a Device Mnemonic

The following options assign a device mnemonic (SYS:, USR:, or GIN:) to a device. The current assignments for these mnemonics are included in the system status information listed by running the SYSTAT utility program. Typing a colon (:) after a device mnemonic is optional. The "=" sign that separates the device mnemonics can be replaced with a single space.

SYS=file-structured device

Assign the mnemonic SYS: to a file-structured device. Generally, the device containing the GOS utility programs is assigned the mnemonic SYS:.

Example

```
#SET SYS=DK
```

Assign the mnemonic SYS: to the device DK: (or DK0:). That hard disc can now be referenced by specifying SYS:, DK:, or DK0:.

USR=file-structured device

Assign the mnemonic USR: to a file-structured device. Generally, the device containing the user's application programs is assigned the mnemonic USR:.

Example

```
#SET USR:=DK1:
```

Assigns the mnemonic USR: to the device DK1:. That hard disc can now be referenced by specifying DK1: or USR:.

GIN=graphic input device

Assign the mnemonic GIN: to a graphic input device (Joystick, Graphics Tablet, or Plotter).

Example

```
#SET GIN=JOY
```

Assigns the mnemonic GIN: to the Joystick. The Joystick can now be referenced by specifying GIN: or JOY:.

Option--Set Date or Time

The following options allow the operator to set the date or time. The current date and time are included in the system status information listed by running the SYSTAT utility program.

DATE day-month-year

Set the current date. Specify only the first three letters of the month and the last two digits of the year. Any ASCII character can be used to separate the day, month, and year, except 0-9, A-Z, and , (comma).

Example

```
#SET DAT 25/JUN/77
```

Set the current date to June 25, 1977.

TIME hours:minutes:seconds [AM or PM]

Set the current time. If "PM" is typed, 12 hours are automatically added to the specified time (converts the time to a 24-hour clock, or "military time"). Any ASCII character can be used to separate the hours, minutes, and seconds, except 0-9, A-Z, and , (comma). Specifying seconds is optional.

Example

```
#SET TIM 11-45-15
```

Set the current time to 11:45:15.

```
#SET TIM 1:45 PM
```

Set the current time to 13:45:00 (using "military time").

Option--Set MVP boundaries

This option allows the operator to define the size of the monitor viewport (MVP) on the Display Monitor. Keyboard input, graphics, listings, and GOS messages are displayed on the MVP.

MVP

Set the boundaries of the monitor viewpoint (MVP). When "SET MVP" is typed and the RETURN key is pressed, a rectangle appears on the Display Monitor. This rectangle represents the boundaries of the MVP. The following message is then displayed on the screen:

Smaller - Function Key 0

Bigger - Function Key 1

Return to original definition - Function Key 2

Terminate definition with any other function key

To decrease the MVP:

Press function key 0 while manipulating the Joystick. The rectangle can be moved without decreasing the size by manipulating the Joystick while not pressing function key 0.

To enlarge the MVP:

Press function key 1 while manipulating the Joystick. The rectangle can be moved without increasing the size by manipulating the Joystick while not pressing function key 1.

To restore the default MVP (full screen):

Press function key 2.

To select the current rectangle on the screen as the MVP boundaries:

Press any other function key (function keys 3-11).

Example**#SET MVP**

The current size of the MVP is outlined by a rectangle on the Display Monitor. The operator manipulates the boundaries of the MVP, then presses any one of the function keys 3-11 to exit the MVP option of the SET utility.

Option--Set Crosshair Cursor Speed or Size

The default graphic input cursor is the crosshair cursor (the graphic input cursor can be redefined under program control). The graphic input cursor is displayed on the screen when graphic input is required. The operator generally manipulates the Joystick to move the graphic input cursor across the screen. These options allow the operator to change the graphic input (crosshair) cursor speed or size.

[CURSOR] SLOW n
FAST n

Set the graphic input (by default, the crosshair) cursor speeds by specifying **n** as an integer value from 1, the slowest speed, to 15, the fastest. The FAST speed is in effect when the graphic input cursor is moved while pressing the SHIFT key (**n** is automatically multiplied by 16). Otherwise, the SLOW speed is in effect. Initially, GOS assigns the value 1 to **n** for both SLOW and FAST speeds.

Example

#SET SLO 3,FAS 8

Set the SLOW graphic input cursor speed to 3 and the FAST graphic input cursor speed to 8.

#SET FAS 15

Set the FAST graphic input cursor speed to 15, the maximum value.

**[CURSOR] SMALL (or LITTLE)
LARGE (or BIG)**

Select the crosshair cursor size. The LARGE crosshair cursor (the default size) intersects the entire viewing area of the screen. The SMALL crosshair cursor intersects a viewing area approximately 5 1/2 cm by 5 1/2 cm in size.

Example

#SET SMA

Select the SMALL crosshair cursor size.

Option--Set Plotter Character Size

This option allows the operator to select the character size used by the Plotter when plotting alphanumeric characters.

[PLO:] CHA n

Set the character size for the Plotter. The size **n** ranges from 1, the smallest character size, to 10, the largest. The default character size is 4.

Example

#SET CHA 7

Set the Plotter character size to 7.

Option--Select Alphanumeric Cursor

The alphanumeric cursor can be a box or low bar that either blinks or remains constantly visible. Initially, GOS assigns the alphanumeric cursor to be a blinking box.

BLINK

Select the blinking alphanumeric cursor.

NOBLINK

Select the non-blinking alphanumeric cursor.

BOX

Select the box (a rectangle) as the alphanumeric cursor.

LOW

Select the low bar (an underline character) as the alphanumeric cursor.

Example

```
#SET NOB,LOW
```

Select the non-blinking low bar as the alphanumeric cursor.

```
#SET BOX,NOB
```

Select the non-blinking box as the alphanumeric cursor.

Option--Assign the Tablet Size to the Tablet Mnemonic

This option allows the operator to assign the small Graphics Tablet or the large Graphics Tablet to the correct Tablet mnemonic. The current assignments for the Tablet mnemonics are included in the system status information listed by running the SYSTAT utility program. Typing a colon (:) after a device mnemonic is optional. The "=" sign that separates the device mnemonic from the Tablet size can be replaced with a single space.

**TB[n] = SMALL (or LITTLE)
 LARGE (or BIG)**

Assign the small Graphics Tablet or the large Graphics Tablet to the device mnemonic TBn: (where n=0 or 1). The device mnemonic of the Graphics Tablet is determined when the device is connected to the 4081. The Tablet's size must be matched with the correct device mnemonic. For example, if the small Graphics Tablet is connected to the 4081 with the device mnemonic TB0:, then the operator must assign the small Graphics Tablet to TB0: (by typing, SET TB0=SMA) before using the small Tablet. Generally, when the GOS program is loaded into memory, the Tablet sizes are automatically assigned to the correct device mnemonics. The operator need assign the size to the device mnemonic by running the SET program only when the connections for the Graphics Tablets have been changed.

Example

#SET TB1=LAR

Assign the large Graphics Tablet to the device mnemonic TB1:. The large Graphics Tablet must be connected to the 4081 with the TB1: mnemonic. An operator or programmer can now reference the large Graphics Tablet by specifying the TB1: mnemonic.

Example

#SET SYS=FD0,USR=FD1,GIN=JOY

Assign SYS: to the Flexible Disc Unit drive FD0:, USR: to the Flexible Disc Unit drive FD1:, and GIN: to the Joystick.

#SET DAT 8-MAY-77,TIM 11:45:15

Set the date to May 8, 1977 and the time to 11:45:15.

#SET USR=FD1,BLI,LOW,CHA 5

Assign FD1: as the USR: device, set the alphanumeric cursor to be a blinking low bar, and set the Plotter character size to 5.

SQUISH

Syntax

SQUISH[;H] device or file[;D]

Defaults

for a device, device: = none

for a file, device: =USR:
 .extension = .LIB

Wild Cards

Wild cards are *not* allowed.

Switches

- ;H Display the SQUISH utility help message.
- ;D Squish only the file structure's directory (not the entire structure). This option can be used when a **Dir Full!** error message is displayed and it is unnecessary to squish the entire structure.

Description

Remove all empty space between files in a file structure by relocating all files toward the beginning of the structure. All empty space is located at the end of the structure. The empty space is now available for the addition of new files. Also, all filename entries in the structure's directory are relocated toward the beginning of the directory allowing new entries to be added to the directory. (Fig. 11-1 illustrates the SQUISH process.)

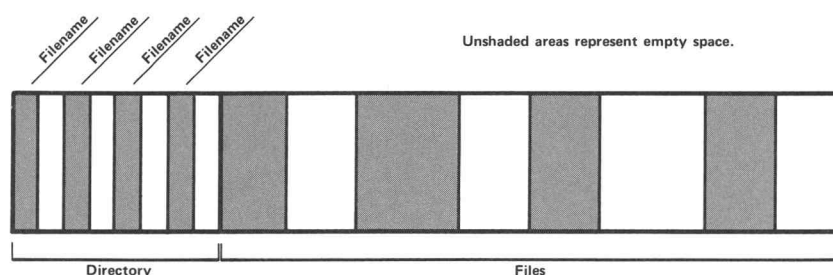
The operator should run the SQUISH utility when any of the following GOS error messages are displayed:

Dir Full! (Directory is full)

Str Full! (File structure is full)

I/O error EM lu n (End of medium)

File structure before SQUISH operation:



File structure after SQUISH operation:

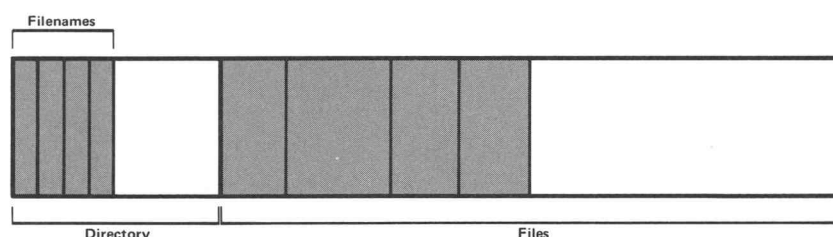


Fig. 11-1. The SQUISH operation.

If the error message is displayed again after the file structure has been squished and an attempt is made to add or append to the file, then there is not enough empty space available in the structure for the addition of new files. In this case, all unnecessary file should be deleted to make room for new files.

Example

#SQUISH USR:

Squish the entire user device.

Dir Full!

#SQUISH FD1::D

While trying to add a file to the flexible disc in FD1:, the error message **Dir full!** was displayed. The operator then ran SQUISH to make more room in the file structure's directory for the new file.

#SQUISH CITY

Squish the entire library file USR:CITY.LIB

Str full!

#SQUISH DK1:YOUR

While trying to add a file to the library DK1:YOUR.LIB, the error message **Str full!** was displayed. The operator then ran SQUISH to make more room in the library file structure for the new file.

SYSTAT

Syntax

SYSTAT [destination device or file]

Defaults

for a destination device, device:=DC:

```
for a destination file,      device:=USR:
                             .extension=.LST
```

Wild Cards

Wild cards are *not* allowed.

Switches

none

Description

Display the current system status information on the screen. If a destination device or file is specified, the system status information is copied to the device or file (in this case, the status information is not displayed). The file size (in blocks) can be specified after the destination file name. If the file size is not specified, the GOS automatic file sizing procedure is used (see **File Sizing** in the section **Files** for more details).

The system status information is displayed in the following format:

System Status of GOS Vx.xLx (Version and level numbers of GOS program)

(Date)

(Time)

Devices: (List of available GOS device drivers, TB[n] size and device assignments for
 SYS:,USR:, and GIN: mnemonics.)

Comm: (Current data communications options for the primary communications
 interface).

(Display Monitor
 character size),

(size of
 memory),

(address of .GOSTOP--
 the lowest memory
 location available
 to the user)

(maximum number of refreshed objects)

Example

#SYSTAT

System Status of GOS V3 0L0

June 27, 1977 12.34.37

Devices: KB,DC,JOY,CT0,CT1,FD0,FD1,FD2,FD3,TB0(Sm),CM0*,PL0
 NUL,SYS(FD0),USR(FD1),GIN(JOY)

Comm 300 Baud, Matched rates, No Parity
 Local Echo, 2 Stop bits, 200ms. Break

Character size 4, 64KB memory, GOSTOP = 6FD0
 Refresh objects = 16

The system status information is displayed on the screen. Currently, this system has device drivers available for:

- the Keyboard Unit (KB:)
- one Display Monitor (DC:)
- the Joystick (JOY:)
- two Cartridge Tapes Unit drives (CT0: and CT1:)
- four Flexible Disc Unit drives (FD0:, FD1:, FD2:, and FD3:)
- one Graphics Tablet (TBO: assigned to the small tablet--see the SET utility)
- one communications interface (CM0:-- the * indicates the primary communications interface)
- one Plotter (PL0:)
- a Null Device (NUL:)

Only these devices can be used in I/O operations.

The mnemonics SYS:, USR:, and GIN: are assigned when the GOS program is loaded into memory or by the SET utility. Currently, the mnemonic assignments are:

- SYS: is assigned to FD0:
- USR: is assigned to FD1:
- GIN: is assigned to the Joystick (JOY:)

The options for the primary communications interface (CM0:) are currently set to:

- 300 Baud
- Matched transmit/receive rates
- No parity check
- Local echo
- 2 stop bits
- a 200 ms. break signal

(See the section **Data Communications in Host Mode** for details on the communications options.)

The Display Monitor character size is 4, the 4081 has 64K bytes of memory, the lowest memory location available to the user (.GOSTOP) is X'6FDO', and up to 16 refresh objects can be displayed at one time on the Display Monitor.

#SYSTAT LP:

Copy the current system status information on the Line Printer. The status information is not displayed on the screen.

#SYSTAT STATE

Copy the current system status information to the file USR:STATE.LST. The status information is not displayed on the screen.

TYPE

Syntax

TYPE[;H] ASCII file[,ASCII file ...]

Defaults

for an ASCII file, device:=USR:
 .extension=.ASM

Wild Cards

Wild cards are allowed.

Switches

;H Display the TYPE utility help message.

Description

Display the contents of an ASCII file on the screen. An ASCII file contains alphanumeric characters, symbols, and special “control” characters stored in binary ASCII code. If two or more files are specified, the files are combined and then displayed.

Press the CTRL-S keys to stop the display of the contents of an ASCII file on the screen. Press the CTRL-S keys again to continue the display. Press the CTRL-O keys to prevent the display of the remaining contents of an ASCII file on the screen. (See **CTRL-O** and **CTRL-S** in the section **Switches, Messages, Modes, Lights, and Keys** for more details.)

Example

#TYPE PROJ1

Display the contents of the ASCII file USR:PROJ1.ASM on the screen.

#TYPE LAUREL.AND,DK1:HARDY

Combine the ASCII files USR:LAUREL.AND and DK1:HARDY.ASM, then display the resulting text on the screen.

#TYPE;H

Display the TYPE utility help message.



SECTION 12

DATA COMMUNICATIONS IN HOST MODE

CONTENTS

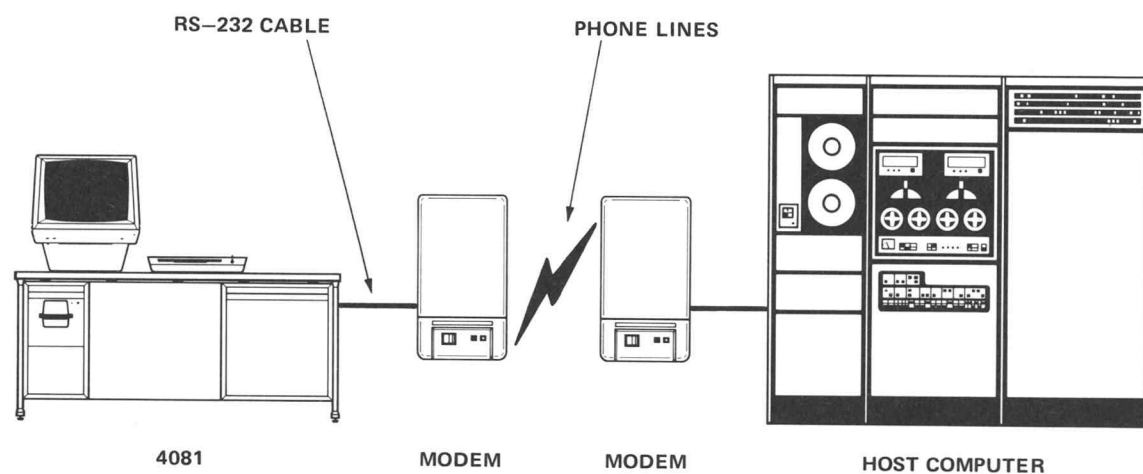
	Page
Communications Concepts	12-4
ASCII Code	12-4
Start and Stop Bits	12-4
Parity Bit	12-4
Baud Rate	12-5
Speeds	12-5
Local and Remote Echo	12-6
Break Signal	12-6
4081 Data Communications Requirements	12-6
RS-232 Compatible Interface	12-7
Serial Asynchronous Data	12-7
Full Duplex	12-7
Running Programs on the 4081 While in Host Mode	12-8
Don't Interrupt the Host	12-8
Initializing Data Communications in Host Mode	12-9
The Communications Interfaces	12-9
Initializing Communications Interfaces	12-10
Setting Data Communications Options	12-10
Selecting the Primary Communications Interface	12-11
Selecting the Standard Options	12-11
Changing the Current Options	12-12
Set Number of Stop Bits	12-12
Set Parity	12-12
Set Baud Rate	12-13
Set Speeds	12-14
Set Echo	12-15
Set Break Signal	12-15
Disabling a Communications Interface	12-16
A Sample Data Communications Session	12-16

Section 12

DATA COMMUNICATIONS IN HOST MODE

When operating in Host mode, the 4081 functions as a full-ASCII alphanumeric terminal. The 4081 must be connected to a host computer, generally through means of a modem and telephone lines (see Fig. 12-1). Information typed on the 4081's keyboard is sent directly to the host computer. Information received from the host computer is displayed on the 4081's Display Monitor.

For special applications, the 4081, while operating in Host mode, can also send data to or receive data from certain external devices such as the Line Printer.



1950-71

Fig. 12-1. Data Communications between the 4081 and a host computer.

COMMUNICATIONS CONCEPTS

When using the 4081 as a terminal, the operator must know the data communication conventions used by the 4081, the modem, and the host computer. To communicate, the 4081, the modem, and the host computer must send and receive data using the same communication conventions.

ASCII Code

Each character transmitted is represented by a sequence of bits (binary digits). Each sequence of bits has a particular meaning depending on the standardized code used to represent a character. For example, the 4081 keyboard transmits characters in sequences of seven bits. The bit pattern for a character is determined by the ASCII code (see the appendix **ASCII Code Chart**). Since the 4081 can only send or receive ASCII code, the host computer must also send and receive ASCII code.

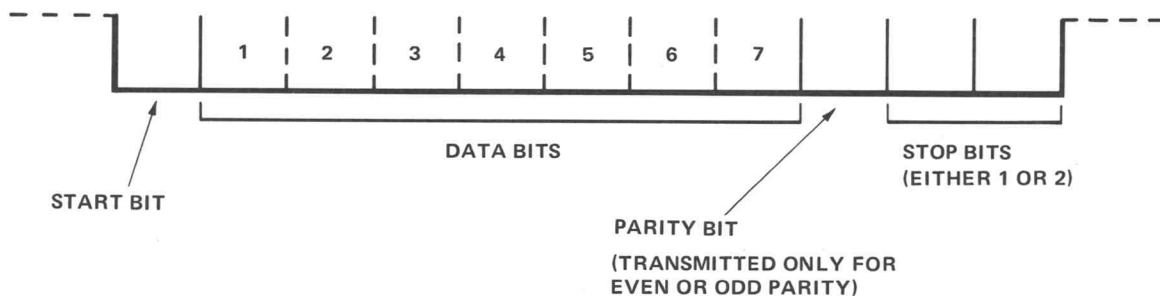
Start And Stop Bits

To separate one character from another, a start bit precedes each ASCII character. One or two stop bits are appended to the end of each character (see Fig. 12-2). The number of stop bits selected depends on the host computer's requirements. The 4081 can send or receive data with either one or two stop bits.

Parity Bit

Before the stop bits, a parity bit can be appended to the ASCII character (see Fig. 12-2). Checking the parity bit allows the detection of errors in transmission between the sender and the receiver. There are two types of parity: even parity and odd parity.

For even parity, the sender counts the number of 1 bits in the ASCII character (the number of bits that are set). If the number of 1 bits is even, then the parity bit is set to 0. If the number of 1 bits is odd, then the parity bit is set to 1 to keep an even number of 1 bits.



1950-72

Fig. 12-2. Format for transmitting a character.

For odd parity, the sender also counts the number of 1 bits in the ASCII character. If the number of 1 bits is odd, however, the parity bit is set to 0. If the number of 1 bits is even, the parity bit is set to 1 to keep an odd number of 1 bits.

When the receiver operates under even parity, it expects that the total number of 1 bits for each character will be even. If the receiver requires odd parity, the total number of 1 bits should be odd. If the parity differs, a transmission error occurred.

The use of the parity bit depends on the host computer. The 4081 can send or receive data with even, odd, or no parity. For no parity, the sender does not transmit a parity bit with each character and the receiver does not check for a parity bit.

Baud Rate

The baud rate measures how many bits per second are transmitted or received. In Host mode, the 4081 can transmit or receive data at 75 to 9600 baud. Generally, baud rates of either 110, 150, 300, 600, 1200, 2400, or 9600 are selected.

The baud rate can be related to the number of characters transmitted or received per second by dividing the baud rate by the number of bits per character. For example, a character is encoded as 1 start bit + 7 data bits + 1 parity bit + 2 stop bits or 11 bits per character (see Fig. 12-2). At 110 baud, 10 characters per second are transmitted ($110/11 = 10$).

Both the 4081 and the host computer must send data at the same baud rate and receive data at the same baud rate. The transmit baud rate does not necessarily have to be the same as the receive baud rate (see **Speeds**). The baud rates selected depend on the capabilities of the modem and the host computer.

Speeds

The 4081 can transmit and receive data at the same or different speeds (baud rates). The operator can select matched speeds, separate speeds, or external speeds. When matched speeds is in effect, the transmit and receive baud rates are the same. The baud rate is set on the 4081.

When separate speeds is in effect, the transmit and receive baud rates are different. The receive rate is set on the 4081. The transmit rate results from dividing the receive rate by a number determined by the Split Rate Divisor strap on the 4081 communications interface. Generally, the Split Rate Divisor strap setting is eight. In other words, the receive baud rate is divided by eight to determine the transmit baud rate (receive baud rate/8 = transmit baud rate). For example, if the receive baud rate is set to 2400, then the transmit baud rate is 300 ($2400/8 = 300$).

When external speeds is in effect, the transmit and receive baud rates are the same. The baud rate, however, is determined by an external clock (either an external clock on the modem or an external clock to the modem). When using the external clock to the modem, the baud rate is set on the 4081. When using the external clock on the modem, the baud rate is determined by the external clock, but the operator must set the same rate on the 4081. In addition, using the external clock on the modem requires a change to the External Clock Source strap on the communications interface. (Contact a Tektronix representative for information on setting the communications interface strap options.)

The decision to select matched, separate, or external speeds depends on the capabilities of the host computer and the modem.

Local And Remote Echo

The 4081 operator can select either local or remote echo. If local echo is selected, characters typed on the keyboard are displayed on the Display Monitor before they're transmitted to the host computer. If remote echo is selected, characters typed on the keyboard are first transmitted to the host computer, then, as the host computer receives each character, the character is sent back to the 4081 and displayed on the Display Monitor.

The decision to select local or remote echo is determined by the capabilities and requirements of the host computer.

Break Signal

When the BREAK key is pressed while the 4081 is in Host mode, a break signal is transmitted. The break signal is actually a sequence of set bits. The sequence is longer than the normal ASCII character. The effect of the break signal is dependent on the host computer or external device that receives it.

The 4081 normally sends a break signal for a duration of 200 milliseconds. Some host computers require a break signal of a longer or shorter duration. The length of time that the BREAK key is depressed does not affect the duration of the break signal.

4081 DATA COMMUNICATIONS REQUIREMENTS

Although the 4081 operator can select stop bits, parity, baud rate, speeds, echo, and break options to conform to the conventions of the modem and host computer, the modem and host computer must also conform to the requirements of the 4081. The 4081 supports full-duplex serial data communications using a RS-232 compatible asynchronous interface. The modem and host computer must also support full-duplex serial RS-232 asynchronous data communications.

RS-232 Compatible Interface

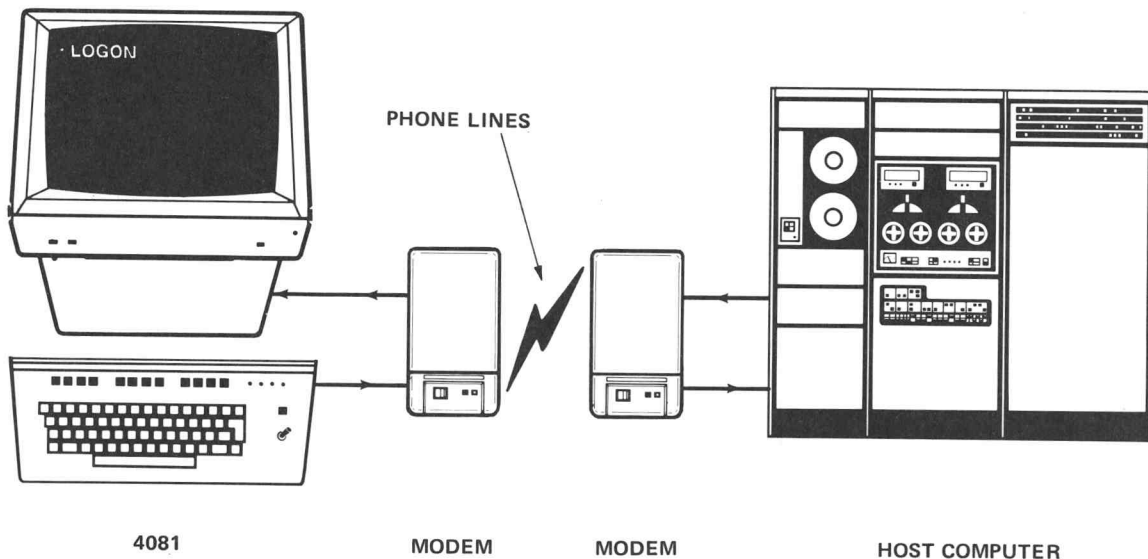
An RS-232 interface conforms to standards developed by the Electronic Industries Association. The 4081 can communicate with host computers and devices that use an RS-232-A or RS-232-C interface. Since the standards for an RS-232 interface allow some options, the following explains the options (serial asynchronous data and full duplex) required by the 4081.

Serial Asynchronous Data

Serial data is transmitted as a string of bits in single file. Asynchronous means that the timing for data transmission is provided by a start bit that precedes each character. The 4081 can only receive or send data formatted in this manner.

Full Duplex

Full duplex means that the 4081 and the host computer are able to send and receive data simultaneously. In other words, the 4081 can send data to the host computer while the host computer sends data to the 4081. The data is transmitted over the same telephone lines at the same time. Full-duplex transmission allows "type-ahead". For example, if remote echo is in effect, the operator types characters on the 4081 keyboard and the host computer transmits the same characters back to the 4081 Display Monitor. Before all the characters are displayed, the operator can type additional characters on the 4081 keyboard ("type-ahead"). The additional characters are sent to the host computer at the same time that previously-typed characters received by the host computer are being sent back to the 4081 for display on the Display Monitor (see Fig. 12-3).



1950-70

Fig. 12-3. Full-duplex data transmission.

RUNNING PROGRAMS ON THE 4081 WHILE IN HOST MODE

To take full advantage of the capabilities of the 4081, the operator can run a program on the 4081 and, while the program is running, the operator can communicate with the host computer in Host mode. For example, the operator runs a batch file on the 4081 that executes several programs. While the batch file is running, the operator presses the SHIFT-BREAK keys to enter Host mode. The batch file continues to execute while the operator communicates with the host computer.

When communications with the host computer are finished, the operator can press the SHIFT-RETURN keys to enter Program mode or the SHIFT-ESC keys to enter Command mode. If the operator enters Program mode, the running program continues to execute without interruption. If the operator enters Command mode, the running program is halted. The operator must then type the resident command CONTINUE to continue execution of the program.

While in Host mode, characters typed on the 4081's keyboard are sent directly to the host computer (see **The Communications Interfaces** in this section for more details). The operator must reenter Program mode to send information from the keyboard to the running program. The 4081's Display Monitor displays information from both the host computer and the running program. The running program, however, can prevent the display of program information by setting a bit in the Job Status Word (see the Plot 80: GOS Programmer's Reference manual for details).

Don't Interrupt The Host

If the running program pauses or exits (usually **Pause** or **Exit** is displayed on the screen) the 4081 enters Command mode, which breaks communication with the host computer. The operator must again press the SHIFT-BREAK keys to reenter Host mode. If the operator does not want host communications to be interrupted, the operator should make sure that either all communications with the host computer are finished before the running program pauses or exits or that the running program will not pause or exit.

One way to prevent a running program from pausing or terminating is to make sure the program requires information from the keyboard before a pause or exit occurs. Because the operator cannot send information from the keyboard to a running program in Host mode, the program must wait until the operator types information on the keyboard in Program mode. In this case, communications with the host computer will not be interrupted by a pause or exit from the running program.

INITIALIZING DATA COMMUNICATIONS IN HOST MODE

Before using the 4081 Graphic System to transmit data to or receive data from a host computer or external device in Host mode:

1. Connect the communications interface's cable to the modem or RS-232 compatible computer communications lines.
2. In Command mode, initialize the communications interface(s). Any changes to the current data communications options should be made at this time.
3. Press the SHIFT-BREAK keys to enter Host mode. The message **<Host>** is displayed on the screen.
4. Log on to host computer.
5. When communications with the host computer are finished, press the SHIFT-ESC keys to enter Command mode or the SHIFT-RETURN keys to enter Program mode.

The Communications Interfaces

Up to six communications interfaces (with device mnemonics CM0:-CM5:) are available. Each of the communications interface can be connected to a different host computer or external device. In Host mode, however, only one communications interface can be used at a given time.

When the 4081 is operating in Host mode, characters typed on the keyboard are transmitted only through the primary communications interface. Any one of the six communications interfaces can be selected as the primary communications interface. Generally, CM0: is the primary communications interface by default. The default primary communications interface is determined by the 4081 user when the GOS IPL tape is created.

NOTE

*The primary communications interface is indicated in the system status information under **Devices:** by an asterisk (*) after the device mnemonic (for example, **CM0 ***). Run the SYSTAT utility program to list the system status information.*

Initializing Communications Interfaces

The operator must initialize a communications interface before it can send or receive data. Initializing a communications interface includes running the SET utility program to assign a set of data communications options to the communications interface. The options (echo, parity, speeds, number of stop bits, baud rate, and break signal duration) determine how data is transmitted and received by the communications interface. The options chosen for a communications interface form a set called the current options.

There is one set of default current options for each communications interface. These are the initial options that are selected according to the needs of the 4081 user when the GOS IPL tape is created.

In addition to the default set of current options for each communications interface, there is one common set of standard options that are also selected when the GOS IPL tape is created. These standard options are, generally, the options most frequently used for data communications. The operator can set the current options for any of the communications interfaces to the standard options.

After initializing a communications interface and assigning it a set of current options, the operator can change any of the current options for the communications interface. The operator's changes remain in effect until the GOS program is again loaded into memory.

NOTE

*The current data communications options for the primary communications interface are listed under the heading **Comm:** in the system status information (run the SYSTAT utility program).*

Setting Data Communications Options

The GOS utility program SET allows the operator to select the primary communications interface, set the current options for a communications interface to the default current options for the interface or to the standard options, and change any of the current data communications options.

NOTE

*At least one of the following SET utility program options must be run to initialize a communications interface for data communications. An exception is the SET utility program option described under **Set Break Signal** in this section. This SET option does not initialize a communications interface.*

Selecting The Primary Communications Interface

To select the primary communications interface, type,

#SET CMn PRI

Select the communications interface CMn: (n is an integer from 0-5) as the primary communications interface. The current options for the communications interface remain in effect. If the communications interface was not previously initialized, then the default current options for the interface are in effect.

Example

#SET CM1 PRI

Select communications interface CM1: as the primary communications interface.

Selecting The Standard Options

To select the set of standard options for a communications interface, type,

#SET CMn

Set the current options for the communications interface CMn: (n is an integer from 0-5) to the set of standard data communications options.

To select the set of standard options for the primary communications interface, type,

#SET COM

Set the current options for the primary communications interface to the set of standard data communications options.

Example

#SET CM2

Set the current options for communications interface CM2: to the set of standard data communications options.

#SET COM

Set the current options for the primary communications interface to the set of standard data communications options.

Changing The Current Options

In some cases, neither the default set nor the standard set of communications options are the desired current options. The operator can change the current options for a communications interface by running any of the following SET options. The current options that have not been modified remain in effect. If the communications interface was not previously initialized, then the default current options for the interface, except the modified options, are in effect.

Set Number of Stop Bits. In asynchronous communications, each character is preceded by a start bit and terminated by one or two stop bits.

To select one stop bit for a communications interface, type,

SET [CMn] ONE

To select two stop bits for a communications interface, type,

SET [CMn] TWO

Select one or two stop bits for a specified communications interface CMn: (n is an integer from 0-5). If no communications interface CMn: is specified, the primary communications interface is assumed.

Example

#SET CM2 TWO

Select two stop bits for data communications through the communications interface CM2:.

#SET ONE

Select one stop bit for data communications through the primary communications interface.

Set Parity. If parity checking and the transmission of a parity bit are not required, the operator can select no parity. If parity checking and the transmission of a parity bit are required, the operator can select either even or odd parity.

To select no parity for a communications interface, type,

SET [CMn] NO

To select even parity for a communications interface, type,

SET [CMn] EVE

To select odd parity for a communications interface, type,

SET [CMn] ODD

Select no, even, or odd parity for a specified communications interface CMn: (n is an integer from 0-5). If no communications interface CMn: is specified, the primary communications interface is assumed.

Example

#SET CM1: NO

Select no parity for data communications through the communications interface CM1:

#SET EVE,CM1 ODD

Select even parity for data communications through the primary communications interface and select odd parity for data communications through the communications interface CM1:.

Set Baud Rate. The baud rate determines the number of bits per second that are transmitted or received.

To select the baud rate for a communications interface, type,

SET [CMn] BAU x

Select the baud rate (x is an integer from 75 to 9600) for a specified communications interface CMn: (n is an integer from 0-5). If no communications interface CMn: is specified, the primary communications interface is assumed.

Example

#SET CM4 BAU 1200

Select a baud rate of 1200 for the communications interface CM4:.

#SET BAU 2400

Select a baud rate of 2400 for the primary communications interface.

Set Speeds. The speeds selection controls the transmit and receive baud rates. If matched speeds is selected, the transmit and receive baud rates are the same. If separate speeds is selected, the transmit and receive baud rates are different. The transmit rate results from dividing the receive rate by a number determined by the Split Rate Divisor strap on the communications interface (generally, the divisor is set to eight).

If external speeds is selected, the transmit and receive baud rates are the same. The baud rate, however, is determined by an external clock on the modem or an external clock to the modem.

To select matched speeds for a communications interface, type,

SET [CMn] MAT

To select separate speeds for a communications interface, type,

SET [CMn] SEP

To select external speeds for a communications interface, type,

SET [CMn] EXT

Select matched, separate, or external speeds for a specified communications interface CMn: (n is an integer from 0-5). If no communications interface CMn: is specified, the primary communications interface is assumed.

Example

#SET CM2 SEP

Select separate speeds for the communications interface CM2:

#SET MAT

Select matched speeds for the primary communications interface.

#SET SEP,BAU 2400

Select separate speeds and a baud rate of 2400 for the primary communications interface. Since separate speeds is in effect, the receive baud rate for the primary communications interface is set to 2400. The transmit baud rate, as determined by the Split Rate Divisor strap on the primary communications interface (in this case, the divisor is set to eight), is 300 ($2400/8=300$).

Set Echo. When local echo is in effect, each character typed on the 4081 keyboard is displayed on the Display Monitor before the character is sent to the host computer. When remote echo is in effect, each character typed on the 4081 keyboard is displayed on the Display Monitor after the host computer receives the character.

To select local echo for a communications interface, type,

#SET [CMn] LOC

To select remote echo for a communications interface, type,

#SET [CMn] REM

Select local or remote echo for a specified communications interface CMn: (**n** is an integer from 0-5). If no communications interface CMn: is specified, the primary communications interface is assumed. Remember that in Host mode, the 4081 keyboard only transmits through the primary communications interface.

Example

#SET CM1 LOC

Select local echo for the communications interface CM1:.

#SET REM

Select remote echo for the primary communications interface.

Set Break Signal. When the BREAK key is pressed while the 4081 is operating in Host mode, the 4081 sends a break signal to the host computer. Generally, the break signal is sent for a duration of 200 milliseconds (the default value). Some host computers require a break signal of a longer or shorter duration.

To set the duration of the break signal for all communication interfaces, type,

SET BREAK n

Set the duration of the signal to **n** milliseconds (**n** is an integer from 50 to 1000). Although the duration of the break signal is set for all communications interfaces, the break signal is only transmitted through the primary communications interface. Remember that setting the break signal duration does not initialize a communications interface.

Example

#SET BRE 300

Set the duration of the break signal sent through all the communications interfaces to 300 milliseconds.

DISABLING A COMMUNICATIONS INTERFACE

Once a communications interface is initialized, it is capable of sending and receiving data until the operator disables it. By disabling a communications interface, the interface is no longer capable of sending or receiving data. However, the current communications options are retained.

To disable a communications interface, execute the GOS resident command ASSIGN to assign the communications interface to a logical unit number. Type,

#ASSIGN lu=CMn:

lu is an integer from 0-15 specifying a logical unit.

n is an integer from 0-5 specifying a communications interface.

Then type the GOS resident command CLOSE. Type,

#CLO

Example

#ASSIGN 5=CM2:

#CLO

Assign the communications interface CM2: to logical unit 5. The GOS resident command CLOSE then disables communications interface CM2:.

A SAMPLE DATA COMMUNICATIONS SESSION

To communicate with a host computer in Host mode, the operator first connects the modem to the communications interface cable. In this sample session, the operator selects communications interface CM0:. By default, communications interface CM0: is generally the primary communications interface.

The operator must then initialize the communications interface. In this session, the default set of current communications options for communications interface CM0: are the options desired for host communications. While the 4081 is operating in Command mode, the operator types:

#SET CM0 PRI

The communications interface is now initialized. The communications interface CM0: is selected as the primary communications interface and the current options are initialized to the default set of current options for communications interface CM0:.

The operator then presses the SHIFT-BREAK keys to enter Host mode and logs onto the host computer.

When the operator is finished communicating with the host computer, the communications interface CM0: should be disabled. The operator types,

```
#ASSIGN 3=CM0:
#CLO
```

Communications interface CM0: is first assigned to logical unit 3. Then executing the resident command CLOSE disables communication interface CM0:. The current communications options for communications interface CM0: are retained.

To continue the session, assume the modem is still connected to primary communications interface CM0:. While the 4081 is operating in Command mode, the operator reinitializes the primary communications interface by running the SET utility program.

To select the standard set of communications options for the primary communications interface, the operator types,

```
#SET COM
```

The current options for the primary communications interface are now initialized to the standard set of data communications options.

The operator then runs the SYSTAT utility program to check the current options for the primary communications interface. The operator types,

```
#SYSTAT
```

and the following is displayed on the Display Monitor:

```
#SYSTAT
```

```
System Status of GOS V3 0L0
```

```
August 22, 1977      10 34 38
```

```
Devices  KB,DC,JOY,CT0,CT1,FD0,FD1,FD2,FD3,TB0(Sm),CM0*,PL0
          NUL,SYS(FD0),USR(FD1),GIN(JOY)
```

```
Comm     300 Baud, Matched rates, No Parity
          Local Echo, 2 Stop bit(s), 200ms Break
```

```
Character size 4, 64KB memory, COSTOP - 6FD0
Refresh objects - 16
```

DATA COMMUNICATIONS IN HOST MODE

As shown after the heading **Devices:**, the device mnemonic CM0 has an asterisk after it, indicating that it is the primary communications interface. After the heading **Comm:**, the current communications options for the primary communications interface are listed. The current options are: 300 baud rate, matched speed of transmit and receive baud rates, no parity, local echo, two stop bits, and a 200 millisecond break signal duration.

The operator, however, knows that odd parity and remote echo are required by the host computer. The operator selects odd parity and remote echo for data communications through the primary communications interface by typing,

```
#SET ODD,REM
```

Odd parity and remote echo are now selected for data communications through the primary communications interface. The other current communications options for the primary communications interface remain in effect.

The operator then presses the SHIFT-BREAK keys to enter Host mode and logs onto the host computer. After completing communications with the host computer, the operator presses the SHIFT-ESC keys to return to Command mode on the 4081. The operator then disables the primary communications interface by typing,

```
#ASSIGN 1=CM0:  
#CLO
```


SECTION 13

GOS ERROR MESSAGES

CONTENTS

	Page
Command Processor Errors	13-3
?.....	13-3
Bad LU!	13-3
Cksm err	13-3
Lod err	13-4
Mem full.....	13-4
Seq err	13-4
Program Execution Errors	13-5
Bad Dev!	13-5
Bad Dir/Parm!	13-5
Dir Full!	13-5
"File" not found!	13-6
File protect error.....	13-6
Locked File!	13-6
No Buf!.....	13-6
Str Full!	13-6
Sys Fu!.....	13-6
DF xxxxyyyy	13-6
FD xxxxyyyy	13-6
Il xxxxyyyy	13-6
PE xxxxyyyy	13-6
Input/Output Errors	13-9
I/O error xx lu n	13-9
Dir I/O error xx lu n.....	13-9

Section 13

GOS ERROR MESSAGES

This section contains a list of GOS error messages, the probable causes for each message, and some solutions. Reloading the GOS program into memory might also help in some situations (insert the GOS IPL tape cartridge in the top Cartridge Tape Unit drive and press the IPL button.) Contact a Tektronix field service representative if the problem persists.

Command Processor Errors

Table 13-1 lists the error messages that are displayed by the GOS command processor when attempting to load or run a program.

Table 13-1
COMMAND PROCESSOR ERRORS

Error Message	Probable Cause
?	Command input error. The command is ignored. Check the command syntax and try again.
Bad LU!	Invalid logical unit number specified. Only logical unit numbers 0 through 15 can be assigned by the user.
Cksm err	<p>Checksum error in program file.</p> <ol style="list-style-type: none">1. An attempt was made to load a program load file that has a bad checksum (bad data). The file must be recreated and stored again. Also, the disc or tape unit might need maintenance.2. An attempt was made to load a program that is not executable (not a program load file).

Error Message	Probable Cause
Lod err	<p>Program cannot be loaded. The problem is in the program.</p> <ol style="list-style-type: none"> 1. A compiled FORTRAN program cannot be loaded or executed directly through GOS. The Linker/Loader must first be used to link modules from the Run Time Library. An executable program load file can be created through the Linker/Loader. (See the PLOT 80: Library Linker/Loader manual for details.) 2. An assembly language program might have any of the following errors: <ul style="list-style-type: none"> ● In a main program, an operand was not specified for the END pseudo-op. (See the PLOT 80: Assembly Language Programming manual.) ● Undefined external references exist. All symbolic references must either be defined within a program or a subprogram linked to the main program. (See the PLOT 80: Library Linker/Loader manual for details on linking subprograms.)
Mem full	<p>Insufficient memory to load program. Try loading the program at a lower address. If the message occurs again, the program is too large. (See the PLOT 80: Library Linker/Loader manual for details on overlays.)</p>
Seq err	<p>Program file sequence error (bad file). This error occurs when a program load file is located in a file structure at a different block number on the medium than specified in the file's directory entry. The file must be recreated and stored again. The operator might also have to reformat the file structure (see FORMAT in the section GOS Utility Programs). Also, the disc or tape unit may need maintenance.</p>

Program Execution Errors

Table 13-2 lists the error messages that are displayed while a program is running on the 4081.

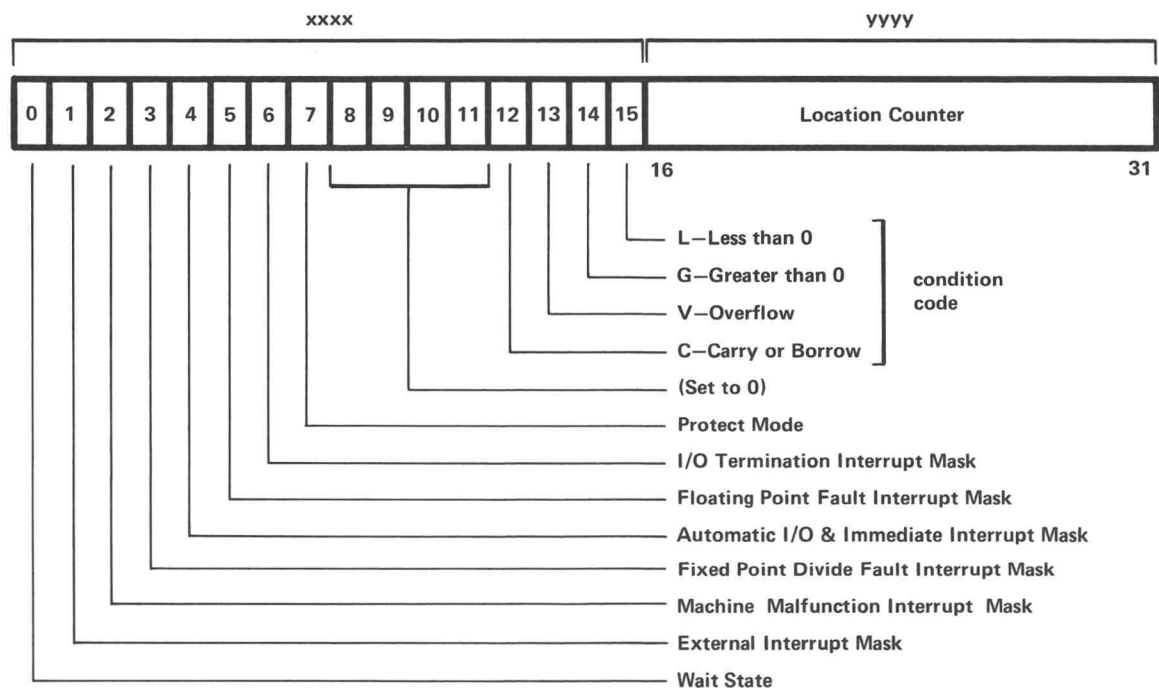
Table 13-2
PROGRAM EXECUTION ERRORS

Error message	Probable Cause
Bad Dev!	Non-existent device. Only the devices contained in the Devices list displayed by running the SYSTAT utility program are available.
Bad Dir/Parm!	<p>Bad directory or invalid program parameters. Try running the DIR utility program. If the Bad Dir! message appears again, the medium must be reformatted (all files are destroyed by reformatting the medium).</p> <p>If invalid parameters is the problem, check the command syntax and required arguments. The problem may be in the program.</p>
Dir Full!	<p>The file structure's directory is full. SQUISH the directory and/or DELETE unneeded files to obtain additional directory space. If the message occurs again, the directory is full and another file structure must be used.</p> <p>In most cases, the Dir Full! error message does not mean that the directory is actually full. The message generally results from the GOS procedure for allocating space in a directory for new file entries.</p> <p>At first, GOS uses only half of the available space in each directory block. When all of the directory blocks have been half-filled with entries and the last directory block has been completely filled, GOS displays the Dir Full! message. To add new files to the file structure, SQUISH the directory (run the GOS utility program SQUISH and specify the ;D switch).</p> <p>When all of the directory blocks are completely full, GOS again displays the Dir Full! message. To add any new files to the file structure at this point, delete unneeded files in the directory to obtain additional space (run the GOS utility program DELETE) and SQUISH the entire file structure (run the GOS utility program SQUISH).</p>

GOS ERROR MESSAGES

Error Message	Probable Cause
"File" not found!	Specified filename not found in directory. Run the GOS utility program DIR to check the names of files contained in the directory.
File protect error	Attempt to open read/write-protected file. To remove read/write-protect attributes from the file, run the GOS utility program ATTRIB without specifying any switches.
Locked File!	Attempt to open read/write-protected file. To remove read/write-protect attributes from the file, run the GOS utility program ATTRIB without specifying any switches.
No Buf!	Insufficient memory to execute program. Execute the resident command CLOSE to release dynamic memory and to close any open files. Try running the program at a lower memory address. If the error occurs again, the program is too large to execute. (See the PLOT 80: Library Linker/Loader manual for details on overlays.)
Str Full!	File structure is full. SQUISH the file structure and/or DELETE unneeded files to obtain additional structure space. If the message occurs again, the structure is full and another file structure must be used.
Sys Fu!	Unable to close a currently opened file. Generally, this error occurs because the device containing the file is unavailable (the power is OFF, the drive door is open, or the medium is removed from the drive).
DF xxxxyyyy FD xxxxyyyy Il xxxxyyyy PE xxxxyyyy	Fixed point divide fault. Floating-point divide fault. Illegal instruction. Parity error. All of these errors cause a "warm start". The GOS version and level number is then displayed on the screen. A "warm start" is similar to the function performed by the resident command CLOSE (dynamic memory is cleared, all logical units are assigned to KB:, etc.) See the PLOT 80: GOS Programmer's Reference manual for more details on "warm start".

Error Message	Probable Cause
<div>DF xxxxyyyy</div> <div>FD xxxxyyyy</div> <div>II xxxxyyyy</div> <div>PE xxxxyyyy</div> <div>(cont)</div>	<p>The PE (parity error) is the only error in this group that can be corrected by an operator. (The other errors result from bad instructions in a program and are explained in the following text.)</p> <p>If a PE error occurs, program execution halts. Reload the GOS program (insert the IPL tape cartridge in the top Cartridge Tape Unit drive and press the IPL button), then try running the program again. If the message occurs again, the 4081 memory might be faulty. Contact a Tektronix field service representative.</p> <p>The DF, FD, and II errors result from illegal instructions or overflow/underflow conditions. xxxxyyyy is the contents of the 4081 processor's 36-bit Program Status Word (PSW) in hexadecimal notation at the time the error occurred. (Fig. 13-1 shows the meaning of each bit in the PSW.)</p>



1950-74

Fig. 13-1. Program Status Word (PSW) format.

Error Message	Probable Cause
DF xxxxyyyy FD xxxxyyyy II xxxxyyyy PE xxxxyyyy (cont)	<p>The xxxx portion is the state of bits 0-15 of the Program Status Word (PSW) when the error occurred. The last hexadecimal digit represents the PSW's condition code. The condition code may or may not be set by the instruction that caused the error. Generally, when a DF or FD error occurs the condition code is set by the arithmetic instruction that caused the error.</p> <p>The yyyy portion is the location counter (bits 16-31) of the PSW when the error occurred. Generally, yyyy specifies the memory location of the next instruction after the instruction that caused the error. For the illegal instruction error, however, yyyy refers to the exact location of the illegal instruction.</p> <p>To locate the instruction causing the problem (assembly language programs): If the program is relocatable (the ORG pseudo-op was not used to set the address at which the program is loaded), subtract the bias (the address at which the program was loaded) from yyyy. The result should be the exact (II error only) or approximate location of the problem. (Refer to an Assembler listing for the program.) For example, if yyyy is X'9854' and the program was loaded at X'8000', the problem is located around location X'1854'.</p> <p>If the program is absolute (the ORG pseudo-op was used to set the address at which the program is loaded), no calculation needs to be performed. The address shown by yyyy should be the exact (II errors only) or approximate address of the problem. (Refer to an Assembler listing for the program).</p> <p>DF errors are caused by fixed-point arithmetic overflow or underflow. FD errors are caused by floating-point arithmetic overflow or underflow. (See the PLOT 80: Assembly Language Programming manual for details on the suspected instruction and the condition code that results.)</p> <p>II errors are generally caused by executing data (by not branching around data blocks). Also, if an operand is not specified for the END pseudo-op of a main program, the II error usually occurs when the program is executed. The yyyy address, in this case, is usually the address of the next memory location after the end of the program.</p>

Input/Output Errors

Table 13-3 lists the error messages that are displayed during data transfer operations.

Table 13-3
INPUT/OUTPUT ERRORS

Error Message	Probable Cause
I/O error xx lu n Dir I/O error xx lu n	<p>Occurs while accessing a device or file. Occurs while accessing the directory of a device or library file.</p> <p>xx=DU--Device Unavailable EF--End of File EM--End of Medium FU--File Unsafe IF--Illegal Function NB--No Block (invalid file structure format) UE--Unrecoverable Error n= logical unit number assigned to device or file in error</p>

DU indicates a device is unavailable (the power is off, the drive door is open, or the medium is incorrectly inserted in the drive).

EF indicates the end of the file (the file is too small). The size of the file must be increased. (See **File Sizing** in the section **Files** for details on specifying the size of a file.)

EM indicates the end of the medium (the disc or tape is full). DELETE any unnecessary files and SQUISH the medium. Try storing the file again. If the error occurs again, another medium must be used.

FU indicates the file is unsafe (a file is open and a DU condition occurred). The file must be recreated and stored again.

IF indicates an illegal function (for example, attempting to write data to a write-protected device).

NB indicates that a non-existent block was referenced on a file structure. The file structure should be formatted again (see **FORMAT** in the section **GOS Utility Programs**).

GOS ERROR MESSAGES

UE indicates an unrecoverable error. A UE error generally occurs when the operator is trying to perform an invalid operation (for example, attempting to display a program load file).

If an input/output error occurs and the **Pause** message is displayed, check to see if **n** is between 0 and 15. If so, execute the resident command LU to display a list of logical unit assignments. The problem can then usually be narrowed down to the device or file assigned to the logical unit number **n**. If the problem is corrected, type CON to continue. Otherwise, type CLO to close any open files. (If the **Pause** message is not displayed, all logical units are automatically reassigned to the keyboard.)

If **n** is greater than 15 (usually 16), the error occurred when GOS was attempting to run a program. GOS always assigns logical unit number 16 to a program load file to be run. The error results because, for some reason, GOS was unable to run the program.

Example

```
#RUN TRIAL
Dir I/O error DU lu 16
#
```

GOS was unable to run the program load file USR:TRIAL.OBJ. The **DU** error means that the device containing the file is unavailable. The operator should check the user device to see if the power is OFF, if the medium is incorrectly inserted in the drive, or if the drive door is open.

SECTION 14
INSTALLATION

CONTENTS

	Page
Site Selection	14-3
Physical Dimensions	14-3
Environmental Specifications	14-7
Power Requirements	14-9
Line Power	14-9
Power Outlet Box	14-10
Power Cord	14-10

Section 14

INSTALLATION

DO NOT UNPACK ANY PART OF THE 4081 GRAPHIC SYSTEM UNTIL A TEKTRONIX REPRESENTATIVE INSPECTS FOR DAMAGE. When the 4081 Graphic System is delivered, call the Tektronix Field Office to set the installation procedure in motion. A Tektronix Field Service Specialist installs the hardware and the software and performs a total system check out.

Since a Tektronix representative installs the 4081 Graphic System, the customer's only responsibilities concerning installation are:

- Site selection (prior to arrival of the equipment).
- Means of transportation to the site.
- Personnel to move the equipment to the site and unpack it under Tektronix supervision.
- Saving shipping materials for possible future relocation.

SITE SELECTION

The following information must be considered in selecting the site for the installation of the 4081 Graphic System.

PHYSICAL DIMENSIONS

The passageways to the site must be large enough to allow passage of the 4081 Graphic System without turning it onto its side or onto its end (Fig. 14-1).

INSTALLATION

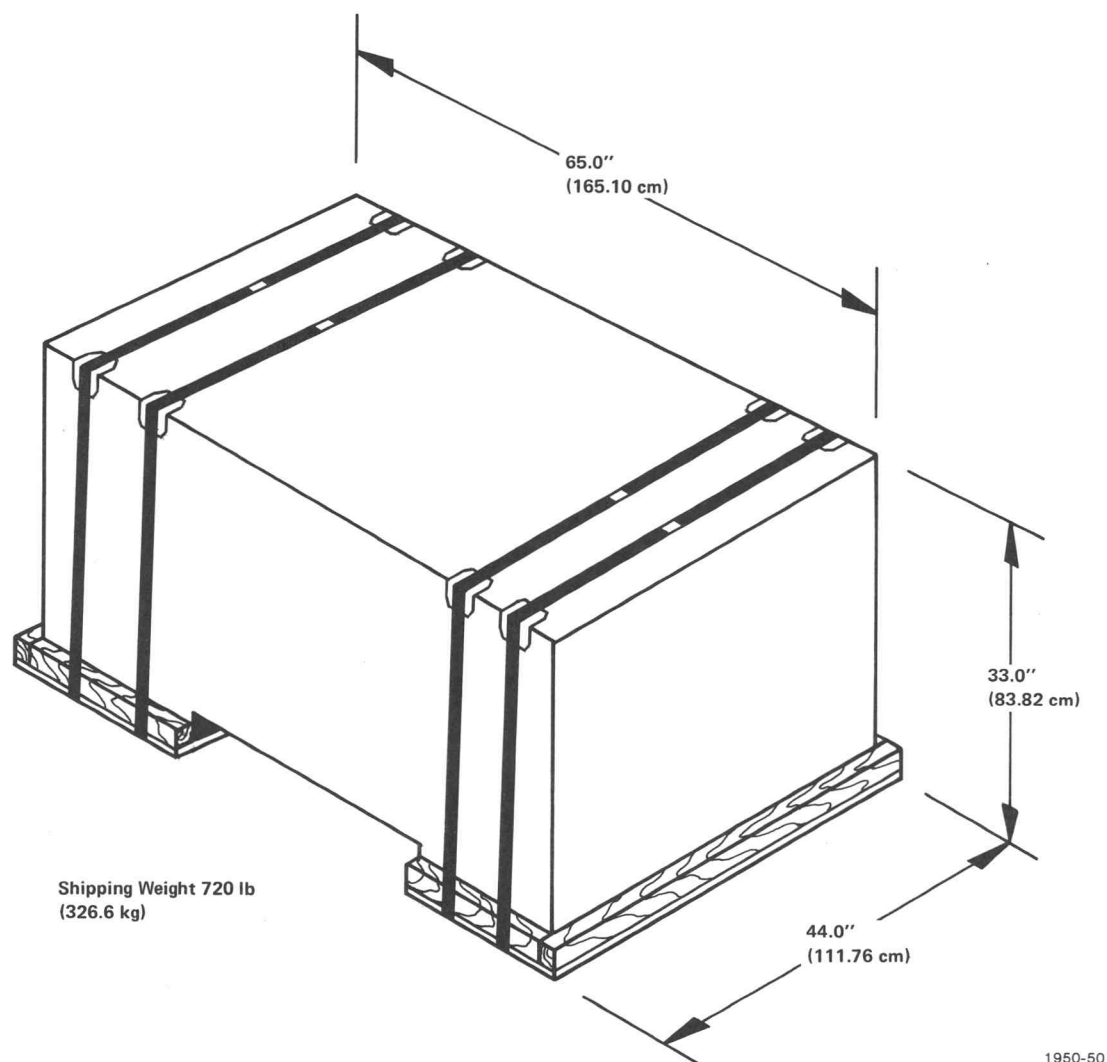


Fig. 14-1. 4081 Graphic System shipping dimensions.

Table 14-1 provides a list of the physical dimensions to consider when deciding where to locate the 4081 (Fig. 14-2).

Table 14-1
PHYSICAL DIMENSIONS

	4081 Graphic System	4081 Graphic System Shipping	4905 Mass Storage Module
Width	60.1 in (152.65 cm)	65.0 in (165.10 cm)	23.1 in (58.67 cm)
Depth	30.0 in (76.20 cm)	44.0 in (111.76 cm)	31.4 in (79.7 cm) w/Flexible Disc Unit 34.0 in (86.45 cm) w/Hard Disc Unit
Height	49.6 in (125.98 cm)	33.0 in (83.82 cm)	27.5 in (69.85 cm)
Weight	655 lb (297.1 kg)	720 lb (326.6 kg)	141 lb to 282 lb (63.9 kg to 127.9 kg)

Minimal workspace is 18 inches (0.45 meters) with unrestricted front and rear maintenance access to the 4081 and allowance for easy access to peripheral devices.

Table 14-1 also provides information for the 4905 Mass Storage Module (Fig. 14-3). It must be placed adjacent to the 4081. An additional 25.6 inches (65.1 centimeters) must be allowed in front of the module for sliding the Hard Disc Unit drive in and out.

WARNING

*Do not slide out more than one 4905 disc drive at a time.
The 4905 can tip over easily if more than one drive is
extended out of the cabinet at the same time.*

INSTALLATION

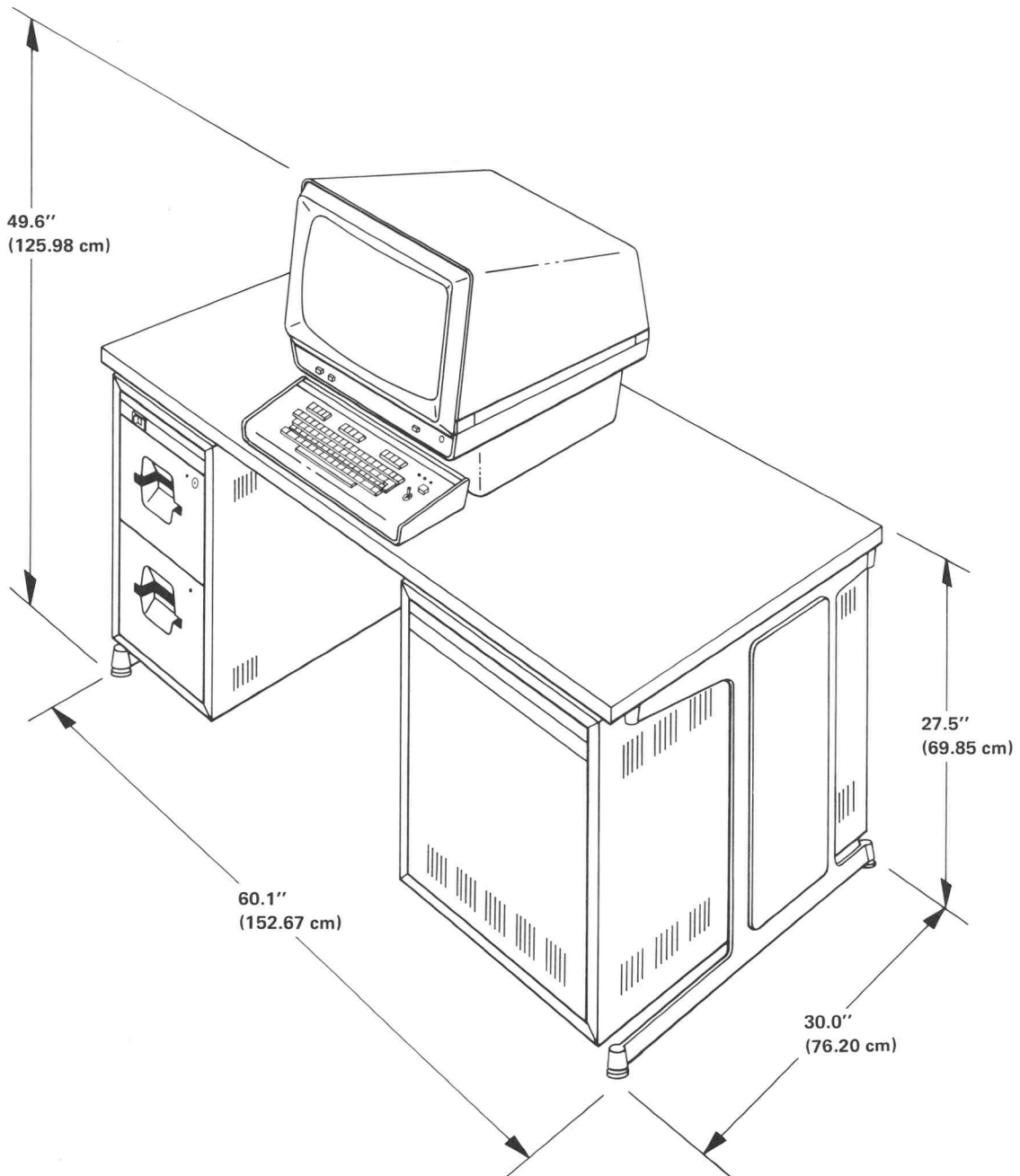
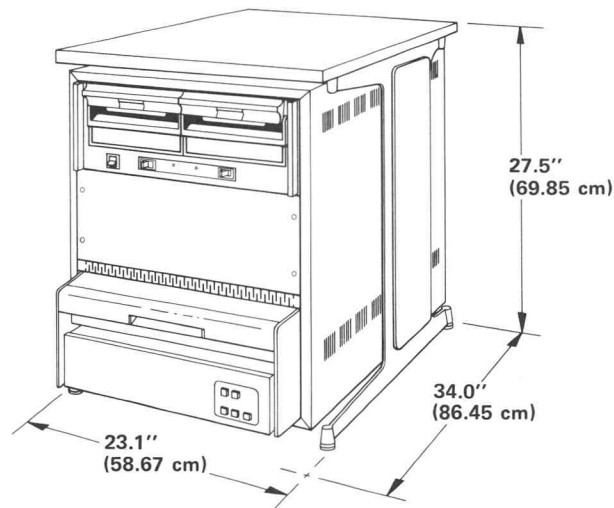


Fig. 14-2. 4081 Graphic System dimensions.



1950-52

Fig. 14-3. 4905 Mass Storage Module dimensions.

ENVIRONMENTAL SPECIFICATIONS

The site selected should be free from dust, dirt, and corrosive fumes. Radiation sources, such as radar or fm transmitters, operating in proximity to the 4081 may affect the operation of the 4081 and certain peripheral devices.

Table 14-2 provides a list of environmental specifications to consider when determining the site for the 4081 Graphic System.

Table 14-2
ENVIRONMENTAL SPECIFICATIONS
(Note exceptions in Table 14-3)

Characteristic	Specification
Temperature Range	
Operating	+ 10 degrees C (50 degrees F) to + 29 degrees C (85 degrees F)
Storage	0 degrees C (32 degrees F) to + 75 degrees C (167 degrees F)
Humidity Range	
Operating	to 70%
Storage	to 90% (noncondensing)
Altitude	
Operating	to 15,000 ft (4572 m)
Storage	to 50,000 ft (15,240 m)
Static Susceptibility	7 kV max, 500 pF
Vibration	
Desk	10-50-10 Hz (cycles per second) 0.010 in (0.0254 cm) Total Displacement
Keyboard	10-55-10 Hz (cycles per second) 0.025 in (0.0635 cm) Total Displacement
Display	20-50-20 Hz (cycles per second) 0.010 in (0.0254 cm) Total Displacement
Heat Dissipation	3450 W (11,782 BTU/hr) maximum
Ventilation	2 in (5.08 cm) minimum clearance on all sides

Table 14-3
TAPE CARTRIDGE ENVIRONMENTAL SPECIFICATIONS

Characteristic	Specification
Temperature Range	
Operating and Storage (with data)	+5 degrees C (41 degrees F) to +40 degrees C (104 degrees F)
Humidity Range	20% to 80% (noncondensing)
Conditioning	The tape cartridge must be conditioned to the operating environment for a time equal to the time away from the environment, not to exceed eight hours.

The 4081 Graphic System is intended for a stationary environment. Its operation and reliability may be affected by excessive vibration.

Optimum reliability is obtained when the operating temperature is below 27 degrees C (80 degrees F) and the humidity is above 40%. Anti-static carpeting also decreases static electricity.

POWER REQUIREMENTS

Line Power

The 4081 Graphic System requires a power source with the following characteristics:

- 115 volts
- 30 amperes
- Single-phase alternating current with safety earth ground
- 60 hertz

The stability of the power source affects the operation and reliability of the system. Maximum recommended power dissipation is 3000 watts at 125 volts and 24 amperes.

INSTALLATION

Operation at 220 volts at 20 amperes requires equipment modification by a Tektronix representative. Refer to the 4081 Technical Data manual for additional information.

Operation with a line frequency of 50 hertz requires software modification of the Graphic Operating System. For more information, consult a local Tektronix representative.

Power Outlet Box

A 2-pole, 3-wire receptacle capable of handling a Hubbell Twist-Lock plug is necessary. Compatible receptacles include Hubbell 2610 and Bryant 70530 FR.

Power Cord

The power cord is 15 feet (4.572 meters) and exits the system behind the Cartridge Tape Unit cabinet (Fig. 14-4).

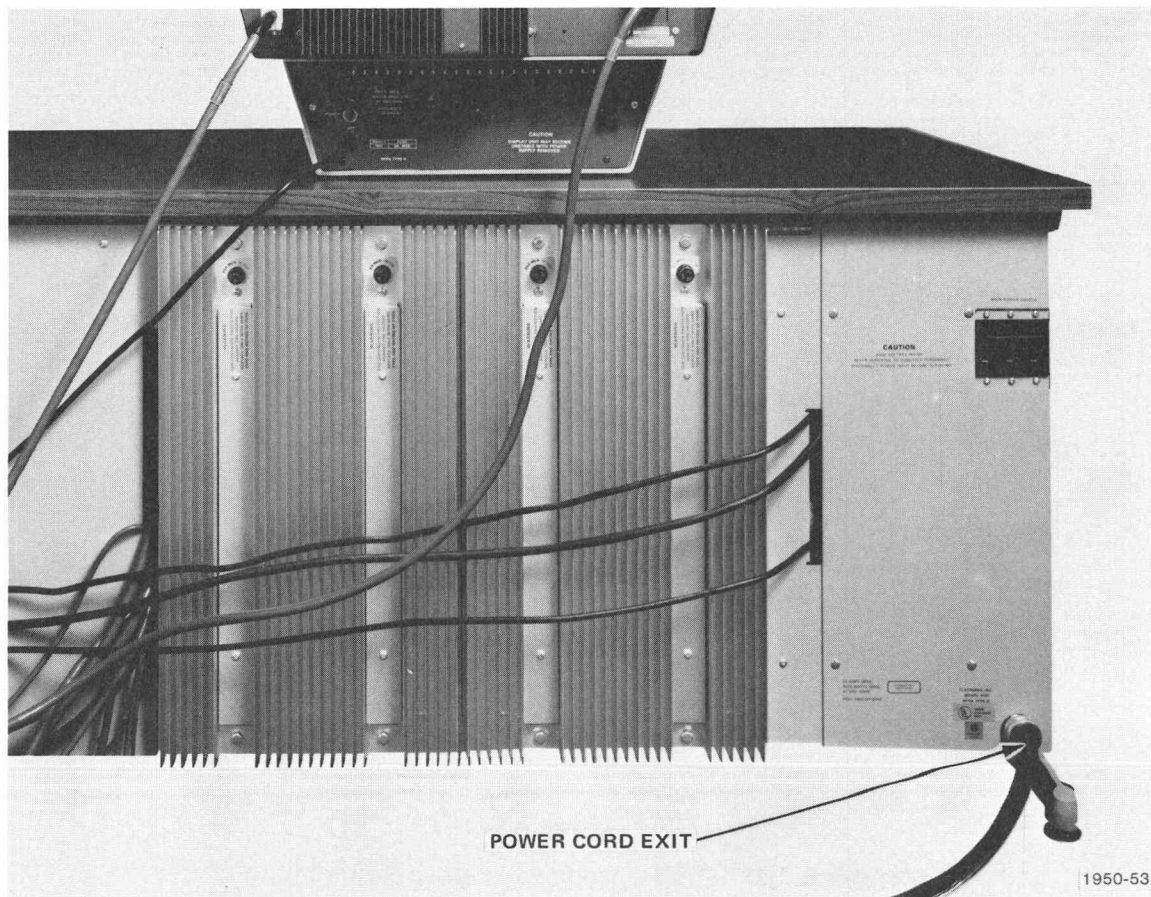


Fig. 14-4. Power cord exit.

If the 4081 Graphic System requires an extension cord, the cord must comply with the requirements in Tables 14-4 and 14-5.

Table 14-4
EXTENSION CORD REQUIREMENTS

Cord Part	Any of the following or the equivalent
Conductor (with UL-listed insulation; see Table 14-5)	No. 10 gauge Cu No. 8 gauge Al
Male Connector (Plug)	Tektronix 131-1381-00 Hubbell 2611 Bryant 70530 NP
Female Connector (Receptacle)	Hubbell 2613 Bryant 70530 NC

Table 14-5
STANDARD POWER CORD CONDUCTOR IDENTIFICATION

Conductor	Color	Alternate Color
Ungrounded (Line)	Brown	Black
Grounded (Neutral)	Blue	White
Grounding (Earthing)	Green-Yellow	Green-Yellow



SECTION 15

MAINTENANCE

CONTENTS

	Page
Routine Maintenance	15-3
Cleaning the Tape Head	15-3
Cleaning the Cooling Vents	15-7
Running Verification Software	15-8
Exterior Surfaces	15-8
Special Maintenance	15-8
Fuse Replacement	15-8
Tape Cartridge Respooling	15-14

Section 15

MAINTENANCE

This section contains routine and special maintenance information for the standard 4081 Graphic System. For maintenance of peripheral devices, see the appropriate documentation (listed in the appendix **Available 4081 Documentation**).

ROUTINE MAINTENANCE

Table 15-1 provides a routine maintenance schedule and lists the only routine maintenance required.

Table 15-1
ROUTINE MAINTENANCE SCHEDULE

Item	Interval
Tape Head	Weekly (more frequently if needed)
Cooling Vents	Monthly
Verification Software	Monthly
Exterior Surfaces	As needed

CLEANING THE TAPE HEAD

Cleaning the tape head in the Cartridge Tape Unit (Fig 15-1) regularly is necessary to prevent data errors and to preserve the life of the tape head. Oxide deposits, dust, and other foreign particles may be deposited on the head during operation and act as abrasives when propelled across the head by tape motion.

Frequency of cleaning depends on the amount of use and the cleanliness of the area in which the unit is used. Cartridge Tape Units with light to moderate use should be cleaned weekly. For heavy use or dusty environments the cleaning frequency should be increased accordingly. Cleaning may be more frequent if data errors occur.

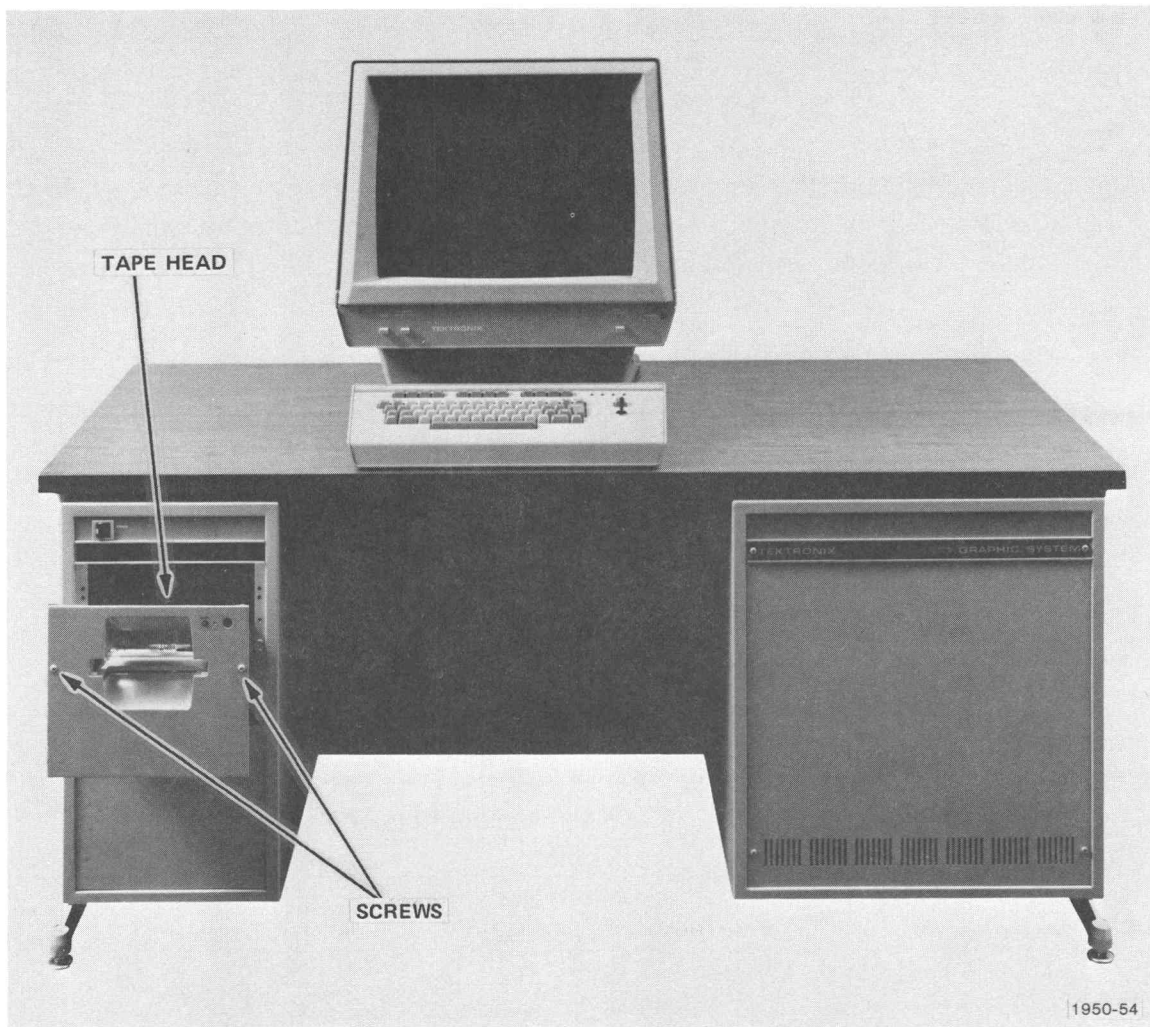


Fig. 15-1. Cartridge Tape Unit drive.

The following procedure describes how to inspect and clean the tape head:

CAUTION

Do not use magnetic devices near the tape head. Do not touch the head with metal or other hard objects. Doing so may damage the head, resulting in damage to tape cartridges and causing a loss of data.

1. Turn off the 4081 Graphic System POWER switch located on the front of the Cartridge Tape Unit cabinet.

WARNING

Dangerous voltages exist inside the 4081. Unplug the 4081 main line cord from its power source before proceeding.

2. Loosen the two screws on the front panel of the Cartridge Tape Unit drive (Fig. 15-1) and slide the drive out of the cabinet.
3. Inspect the tape head (Fig. 15-2) by shining a light, such as a penlight, at an angle across the surface of the head. This reveals accumulated matter or damage to the head.
4. If the head is dirty, continue with this procedure. However, if the head is scratched, scored, or excessively worn (Fig. 15-3), it should be replaced by a Tektronix Field Service Specialist.

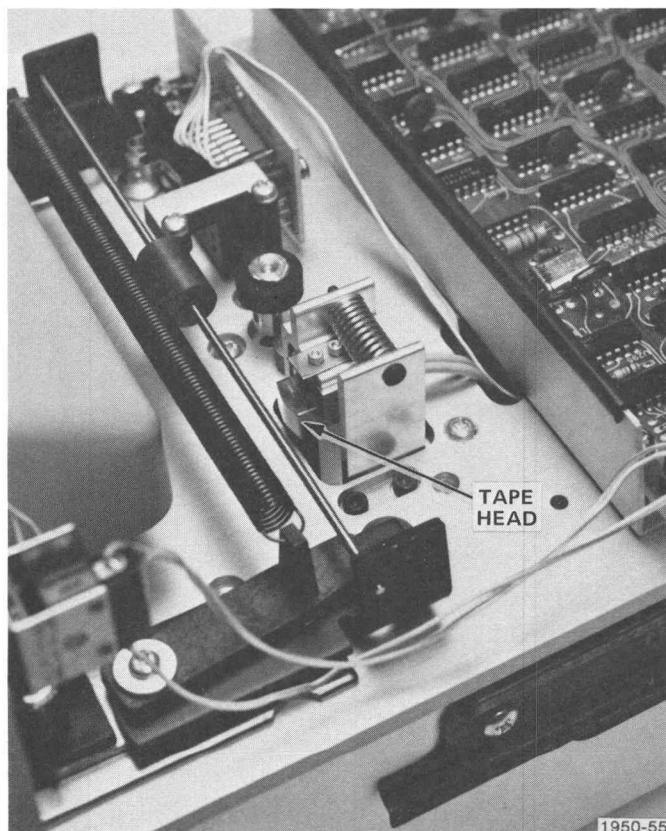


Fig. 15-2. Tape head location.

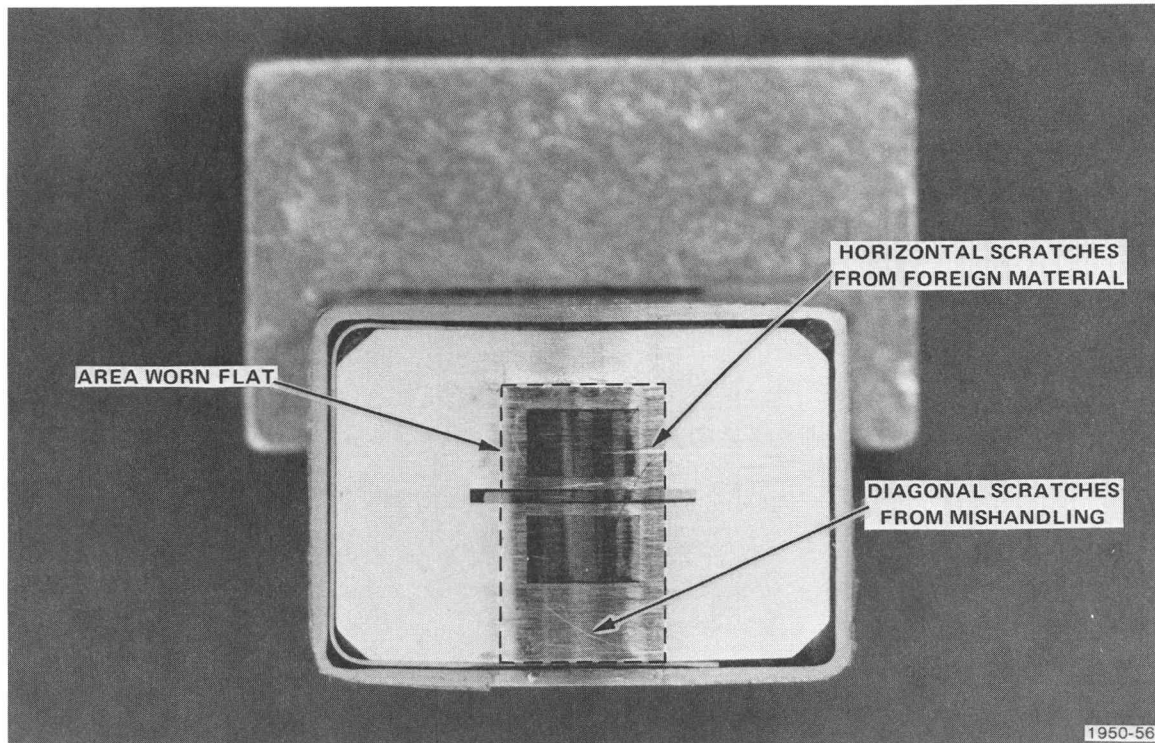


Fig. 15-3. Tape head damage.

5. To rub off accumulated matter, use a cotton swab or special cleaning pad (available through a Tektronix representative) moistened with isopropyl alcohol. Light accumulations of oxide are usually readily removable. Heavy or long-term accumulations may require more cleaning with alcohol and clean swabs. Use extreme care when cleaning the head to prevent scratching or damaging the head surface.
6. After removing all accumulated matter, use a clean, dry cotton swab to polish the head and remove alcohol residue.
7. Slide the Cartridge Tape Unit drive back into the cabinet and tighten the two screws on the front panel.
8. Plug in the 4081 line cord.

CLEANING THE COOLING VENTS

Items such as pieces of paper and balls of dust that lodge beneath the right section of the desk cabinet (Fig. 15-4) can interfere with the cooling system. The bottom panel of this section contains many tiny holes that act as cooling vents. Turn off the POWER switch on the front of the Cartridge Tape Unit cabinet before vacuuming or dusting these vents and the surface beneath the cabinet.

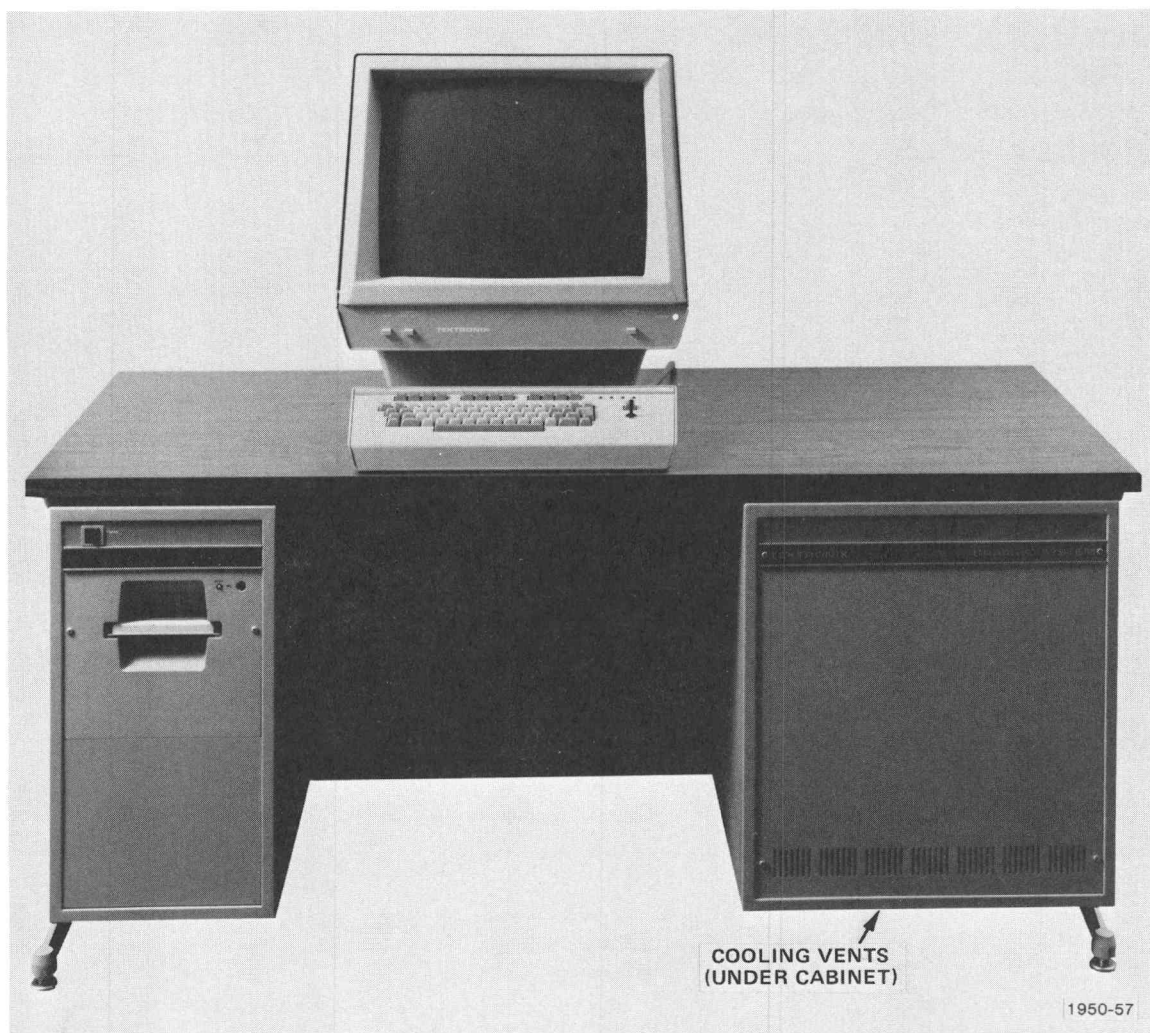


Fig. 15-4. Cooling vents.

RUNNING VERIFICATION SOFTWARE

Refer to the 4081 Verification Program manual for information on running the 4081 Verification Program.

EXTERIOR SURFACES

The screen of the Storage Display Monitor can be cleaned with a glass cleaner. The Keyboard Unit, the other surfaces of the Display Monitor, and the sides of the desk unit can be cleaned with a soft cloth dampened with a solution of mild detergent and water. Commercial formica cleaner and polish are adequate for cleaning and preserving the finish of the desk top.

CAUTION

Avoid the use of chemical cleaning agents that might damage the plastics, paint or metal used in this instrument. Avoid chemicals that contain benzene, toluene, xylene, acetone or similar solvents.

Touch-up paint for extensive scratches and finish damage may be ordered through a Tektronix representative.

SPECIAL MAINTENANCE

FUSE REPLACEMENT

WARNING

Dangerous voltages exist inside the 4081. Unplug the 4081 main line cord from its power source before removing any fuse.

An operation failure may indicate that a fuse needs replacing. If the keyboard prompt lights (Fig. 15-5) fail to operate, check the power supply fuses on the rear of the desk cabinet (Fig. 15-6). The fuses labeled 25A MED 5V cannot be inspected visually. They can be removed from the holder and tested with an ohmmeter or by a Tektronix Field Service Specialist.

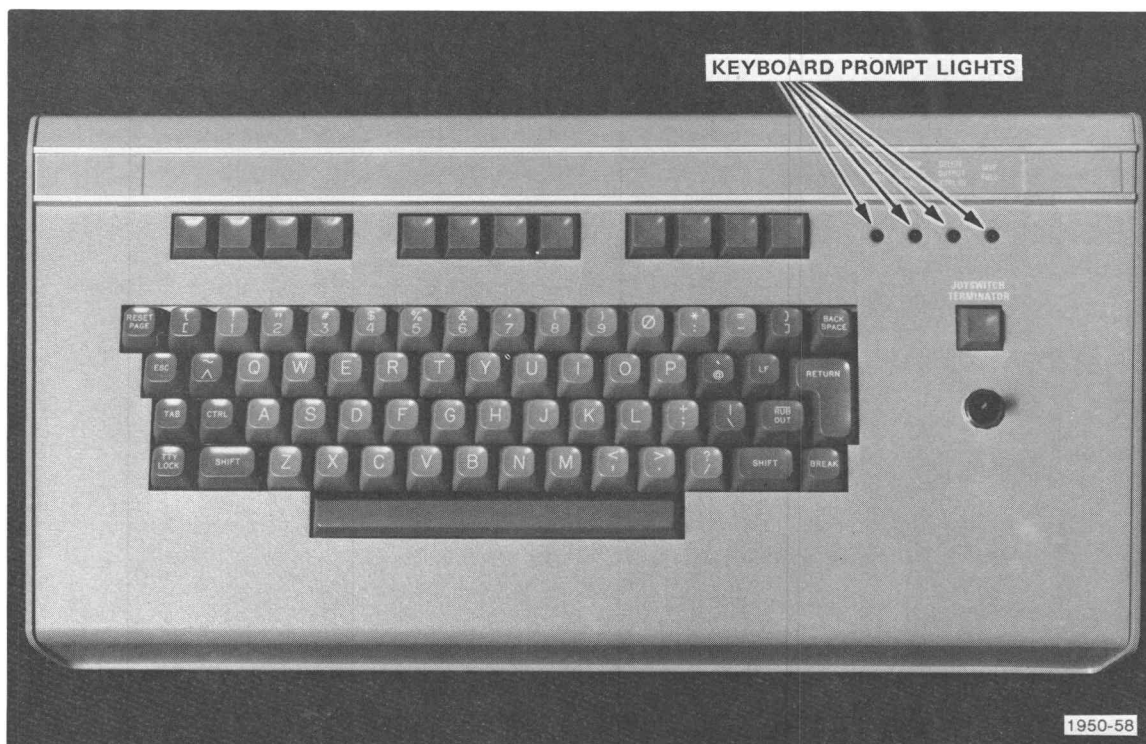


Fig. 15-5. Keyboard prompt lights.

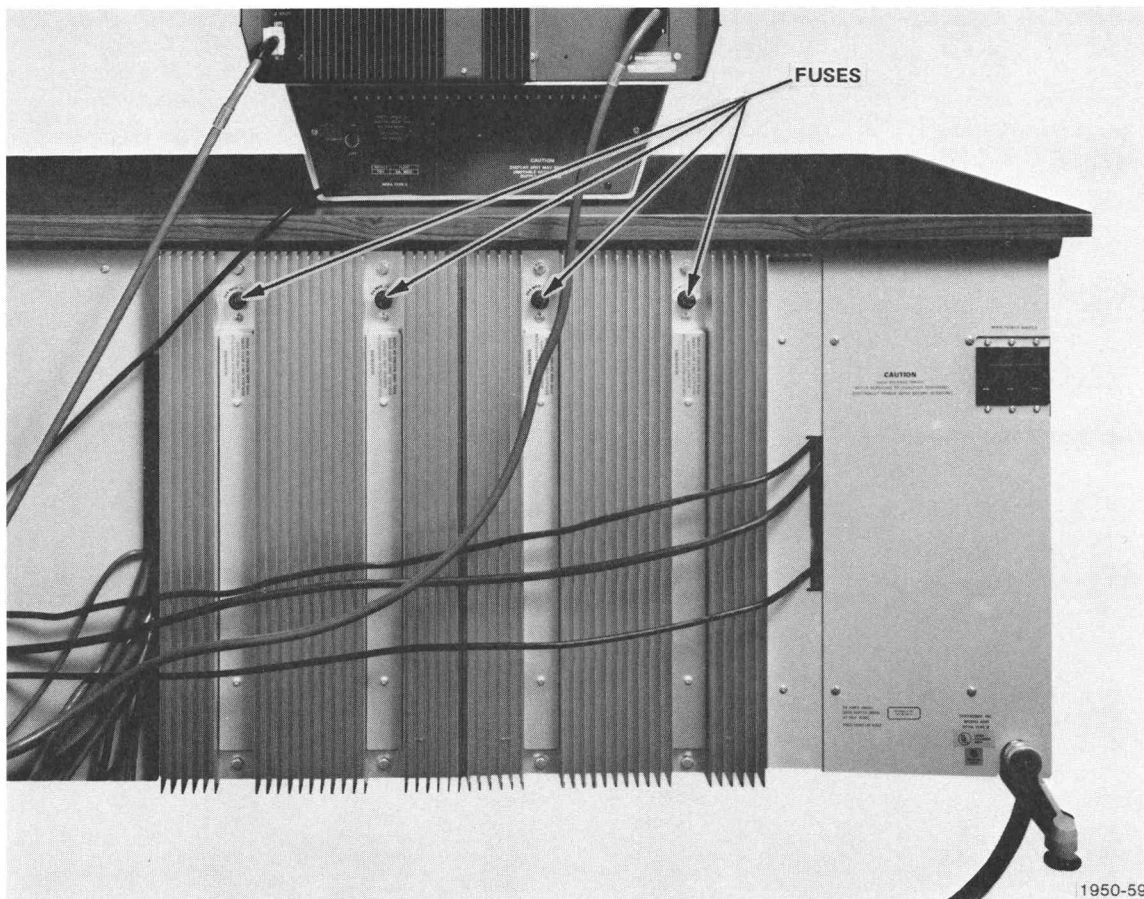


Fig. 15-6. Power supply fuses.

If the Storage Display Monitor fails to light, check the fuse on the back of the Display Monitor (Fig. 15-7).

If the 4081 Graphic System does not IPL, remove the rear panel from the desk cabinet (as described in **The System Power Bus** in the section **Operating Considerations**) and check the Cartridge Tape Unit fuse located directly above the power plug on the rear of the Cartridge Tape Unit cabinet (Fig. 15-8).



Fig. 15-7. Storage Display Monitor fuse.

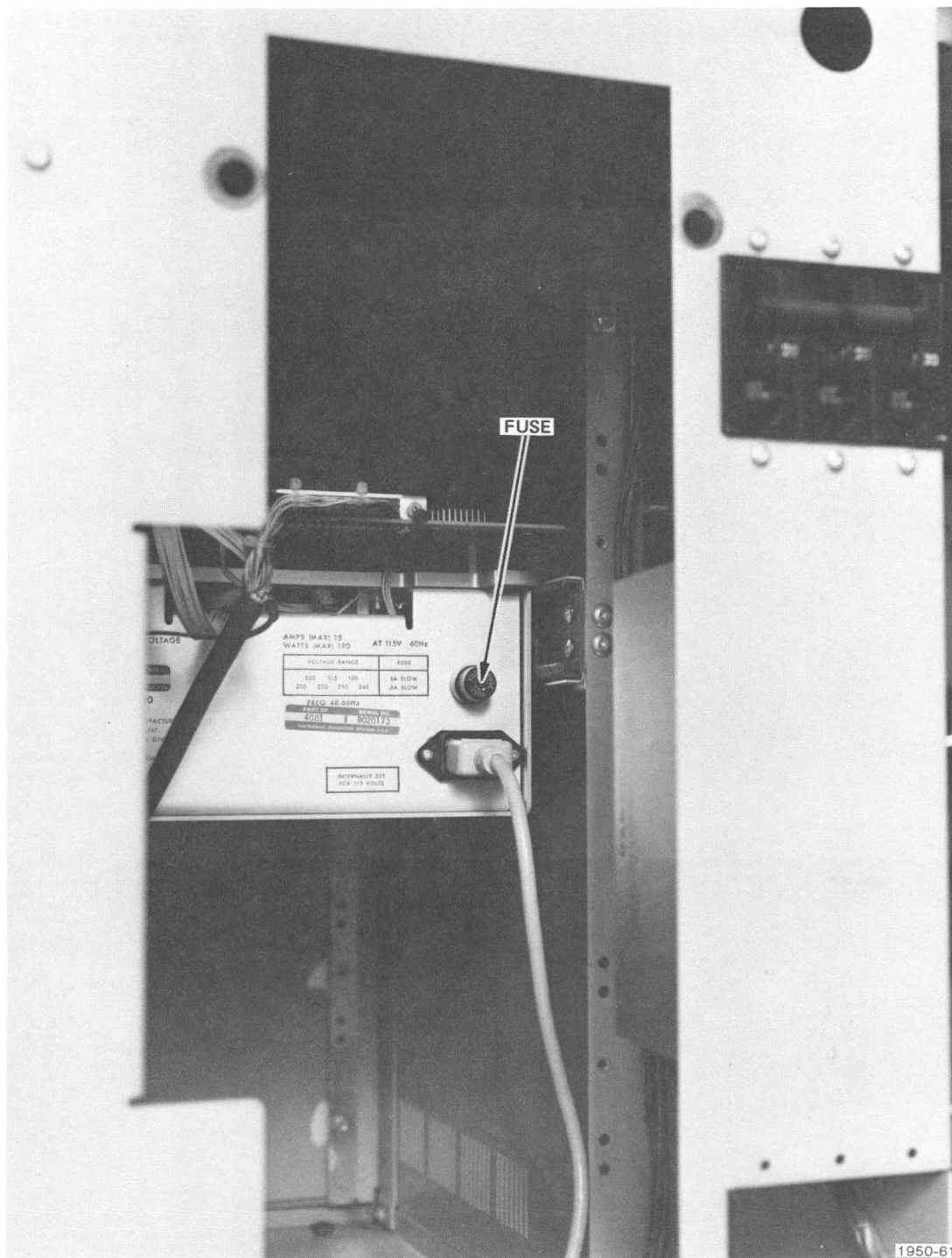


Fig. 15-8. Cartridge Tape Unit fuse.

The following procedure describes how to replace any of the 4081's fuses:

1. Turn off the 4081 Graphic System POWER switch located on the front of the Cartridge Tape Unit cabinet.
2. Unplug the 4081 line cord.
3. Turn the fuse holder 1/4 turn counterclockwise.
4. Pull the fuse holder straight out.
5. Replace the fuse in the fuse holder.
6. Replace the fuse holder and gently turn it clockwise until it slips into a groove.
7. Push the fuse holder straight in (against spring loading).
8. Turn the fuse holder 1/4 turn clockwise.
9. Plug in the 4081 line cord.
10. Turn on the 4081 Graphic System POWER switch.

If a fuse blows repeatedly, do not continue to operate the 4081. Call the Tektronix Field Office. Table 15-2 lists the values for correct fuse protection.

Table 15-2
FUSE PROTECTION

Fuse	115 Vac	220 Vac	+5 Vdc
Power Supply	8 A med blow	5 A med blow	25 A med blow
Cartridge Tape Unit	1 A slow blow	0.6 A slow blow	
Storage Display Monitor	5 A med blow	3 A slow blow	

TAPE CARTRIDGE RESPOOLING

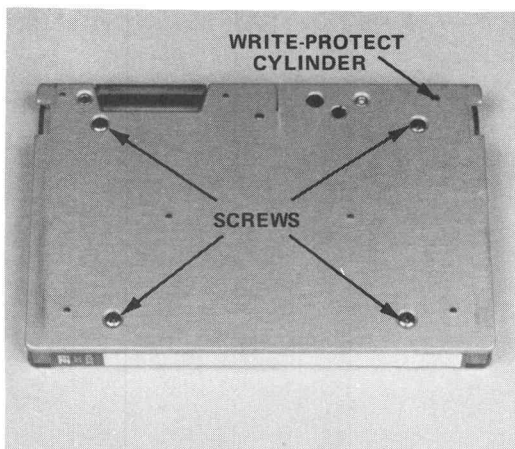
The tape cartridge is open-ended; that is, the tape ends are not secured to either of the spools. Light sensing of small holes near each end of the tape stops tape motion before the physical end of the tape is reached. The tape may, however, occasionally run off one of the spools.

Since tape positioning is critical to proper interpretation of data, it might not be possible to restore a tape that has run off a spool. If it is impossible to recover data from a tape that has been respooled, the tape can be reformatted to record new data (see **FORMAT** in the section **GOS Utility Programs**). The following procedure describes how to respool a tape cartridge:

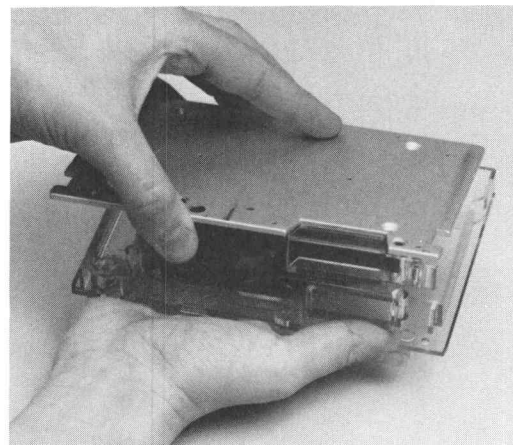
CAUTION

Do not use a magnetic screwdriver when working on or around a tape cartridge.

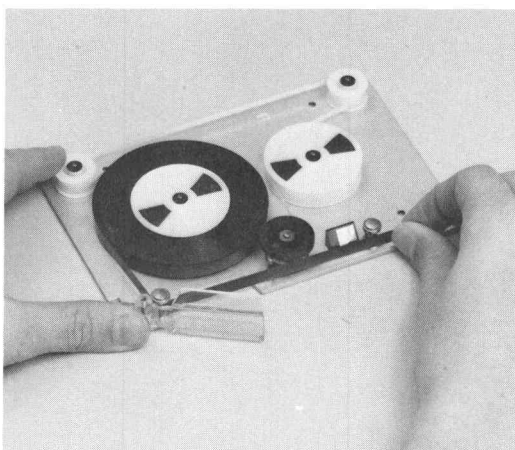
1. Position the tape cartridge with the metal side up and remove the four screws that attach the metal base to the plastic cover (Fig. 15-9a).
2. Carefully remove the cartridge base from the plastic case (Fig. 15-9b). Be careful not to lose the plastic write-protect cylinder or the small metal spring washer between the cylinder and the metal base.
3. Turn the base over and place the loose end of the tape across the front of the cartridge, threading it in front of the two guide posts (Fig. 15-9c).
4. Keeping light tension on the tape, place the loose end of the tape around the outside edge of the take-up spool to the point where the spool meets the tension band (Fig. 15-9d). Rotate the drive roller (Fig. 15-9e) clockwise, causing the tape to pass around the take-up spool, with the loose end passing between the tension band and the spool.
5. Hold the loose end of the tape against the spool and rotate the drive roller until the loose end passes under the continuing length of tape. Continue to rotate the spool by turning the drive roller until three sets of double holes have passed both guide posts. Make certain that these first windings are centered between the spool edges.



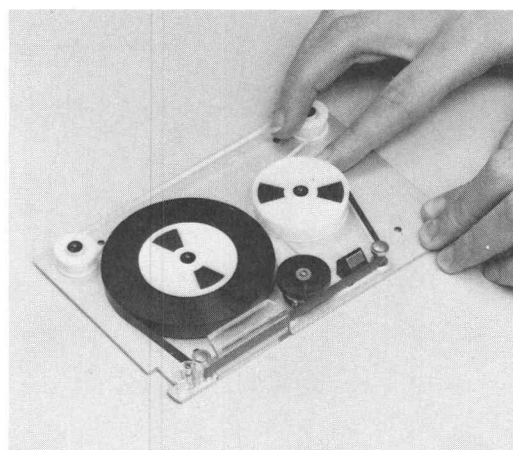
A. Positioning the tape cartridge.



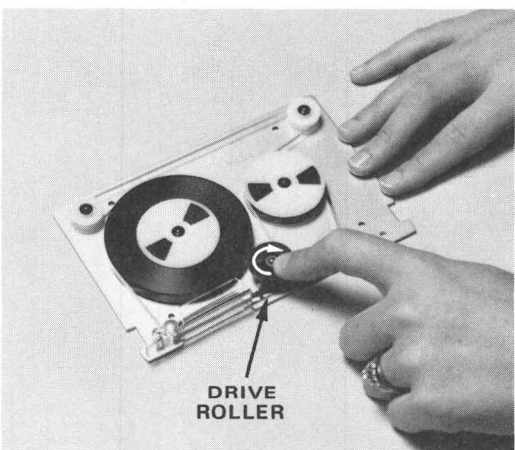
B. Disassembling the tape cartridge.



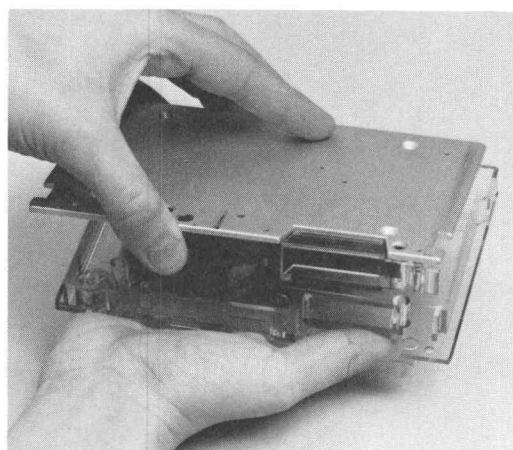
C. Tape positioning within the cartridge.



D. Beginning the tape winding.



E. Winding the tape.



F. Assembling the tape cartridge.

1950-62

Fig. 15-9. Tape cartridge respooling procedure.

MAINTENANCE

6. Be sure that the write-protect cylinder is in position, with the spring washer between the cylinder and the metal cartridge base. Turn the base over and carefully position it into the plastic case, making certain to fit the write-protect cylinder through the opening in the plastic case (Fig. 15-9f). Be careful not to catch or wrinkle the tape with the plastic case.
7. Holding the cartridge together, replace the four screws that attach the metal base to the plastic case.

Several conditions can cause a tape to run off one of the spools. If a subsequent tape runs off a spool, do not load another tape cartridge. Call the Tektronix Field Office.

SECTION 16

OPERATING CONSIDERATIONS

CONTENTS

	Page
The System Power Bus.....	16-3
Tape Cartridge Care	16-7
Storage Display Monitor Controls	16-8
FOCUS.....	16-8
WRITE THRU INTENSITY.....	16-10
HARD COPY INTENSITY	16-11

Section 16

OPERATING CONSIDERATIONS

THE SYSTEM POWER BUS

The circuit breaker on the back of the desk cabinet provides overload protection for the 4081 Graphic System and all peripheral devices plugged into the system power bus. Fig. 16-1 shows the location of the circuit breaker and the system power bus.

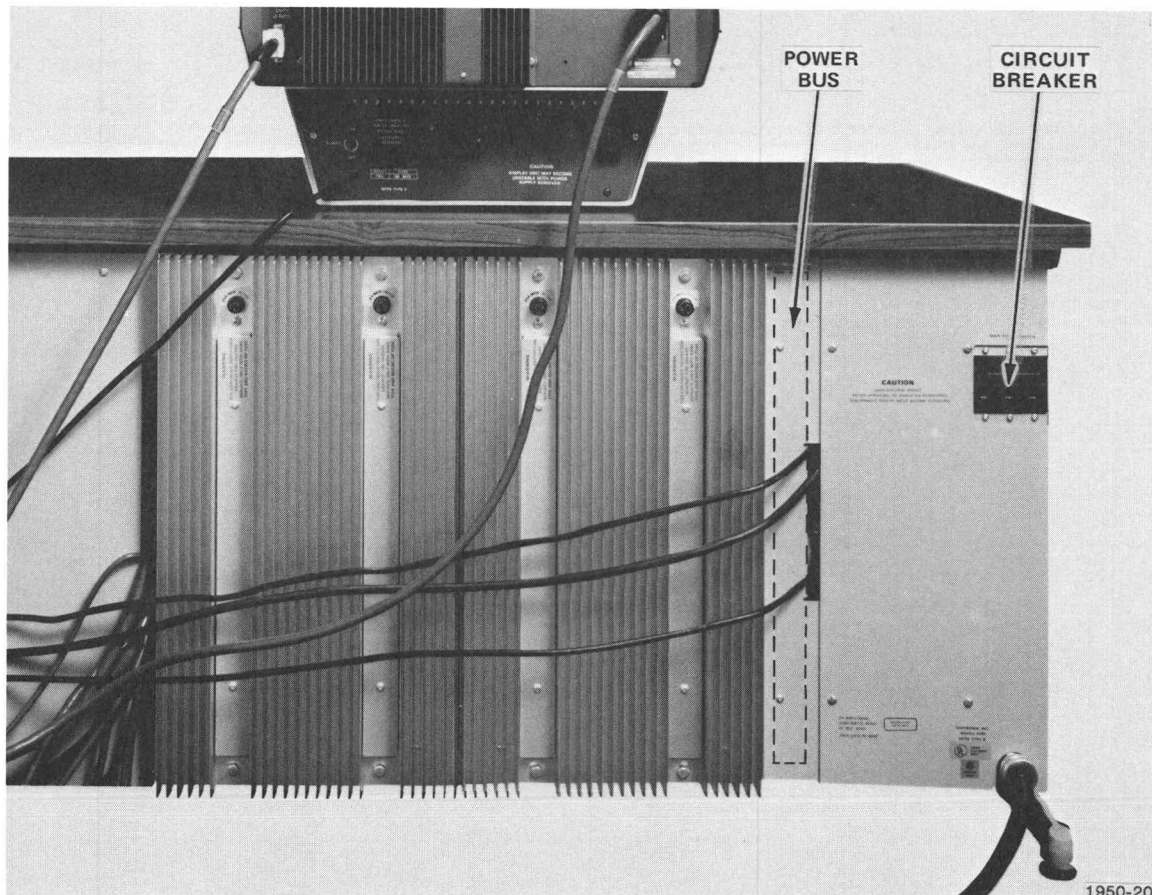


Fig. 16-1. Circuit breaker and system power bus.

OPERATING CONSIDERATIONS

Fig. 16-2 shows the top and bottom halves of the system power bus. The 30 ampere power line is divided to provide 20 ampere service to each half, but the *total* current must not exceed 30 amperes (including the normal amounts drawn by the power supply, the Cartridge Tape Unit, and the Storage Display Monitor). Maximum recommended current is 24 amperes for 115 volt operation.

If the circuit breaker trips, check for excessive loading on the system power bus. Pull the circuit breaker up to restart the system. If it continues to trip, unplug the system and call the Tektronix Field Office.

Table 16-1 lists the power requirements for a number of peripheral devices (to assist in distributing the load equally between the halves of the power bus). These requirements are for 115 volt operation at a line frequency of 60 hertz. (Divide each value by two for 220 volt operation.)

Table 16-1
PERIPHERAL DEVICE POWER REQUIREMENTS

Device	Max Current	Nominal Current
Main Power Supply		5.5 A
Secondary Power Supply		5.5 A
Display Unit		3.0 A
Cartridge Tape Unit		0.9 A
Flexible Disc Unit		
Two drives		2.6 A
Four drives		5.2 A
Hard Copy Unit	12 A	0.6 A
Character Printer	3 A	
Graphics Tablets		
Small Tablet		0.3 A
Large Tablet		0.3 A
Digital Plotter	0.75 A	
Hard Disc Unit		
One drive	5 A	2.3 A
Two drives	10 A	4.6 A

For additional information, see the documentation for the specific peripheral device (listed in the appendix **Available 4081 Documentation**).

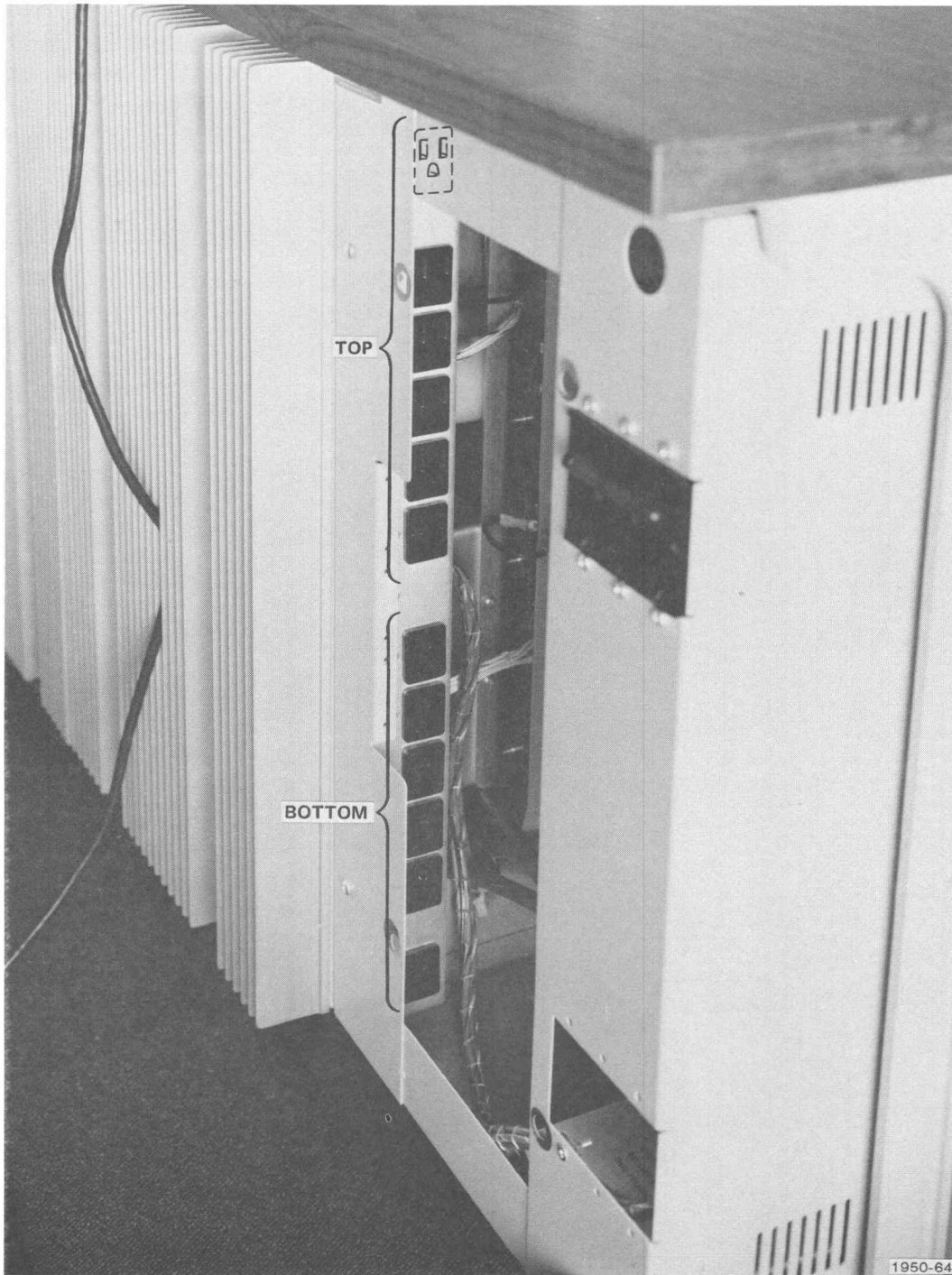


Fig. 16-2. System power bus halves.

OPERATING CONSIDERATIONS

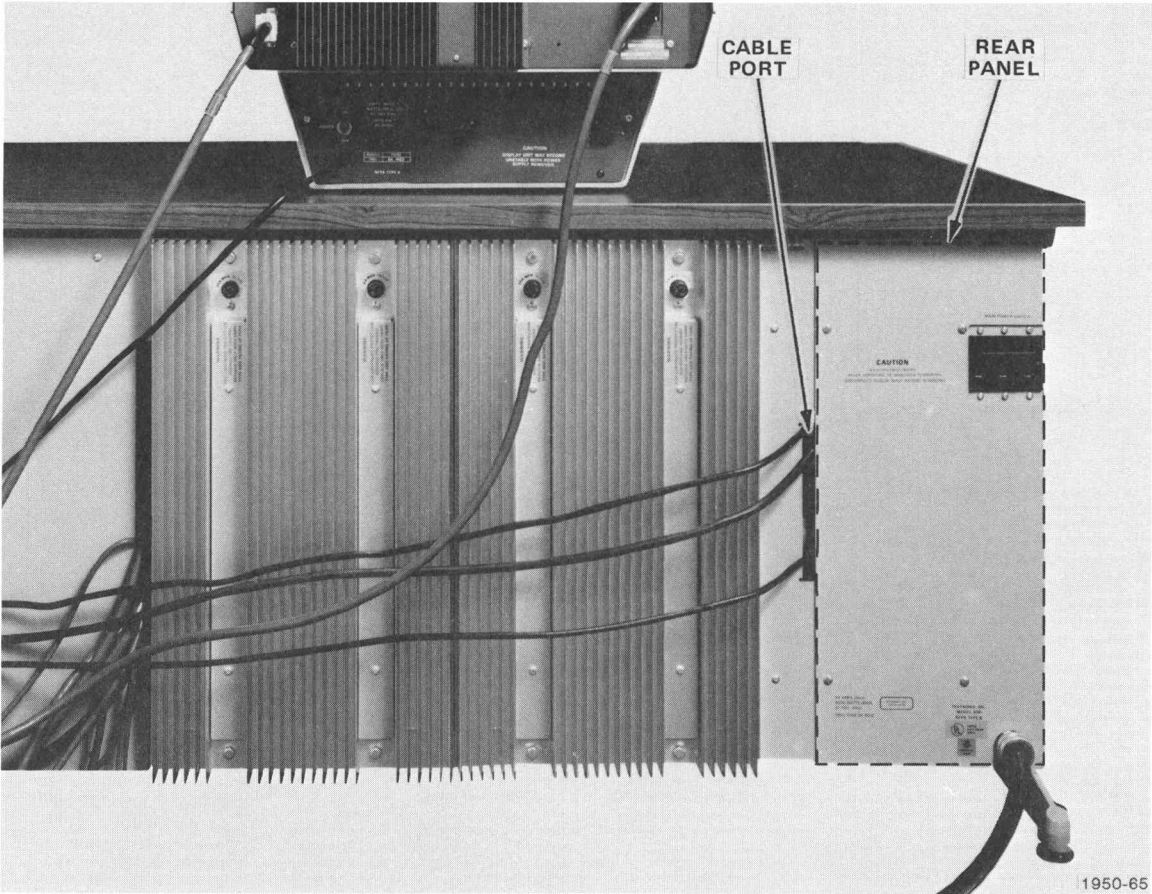
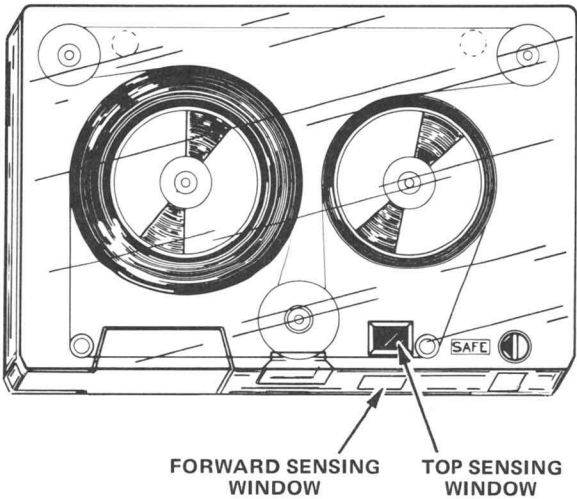


Fig. 16-3. Rear panel and cable port.



(1908) 1950-66

Fig. 16-4. Tape cartridge components.

The following procedure describes how to plug a peripheral device (such as the Plotter or an electric light) into the system power bus:

1. Turn off the 4081 Graphic System POWER switch located on the front of the Cartridge Tape Unit cabinet.

WARNING

Dangerous voltages exist inside the 4081. Unplug the 4081 main line cord from its power source before proceeding.

2. Locate the rear panel of the desk cabinet (Fig. 16-3).
3. Turn each of the four large screws 1/2 turn counterclockwise.
4. Pull the panel straightaway from the rear of the instrument.
5. Plug the device's power cord into the system power bus.
6. Exit the cord through the cable port (Fig. 16-3).
7. Replace the rear panel of the desk cabinet.
8. Turn each of the four large screws 1/2 turn clockwise.
9. Plug in the 4081 line cord.
10. Turn on the 4081 Graphic System POWER switch.

TAPE CARTRIDGE CARE

Performance of the Cartridge Tape Unit and prevention of data errors are partially dependent on the handling of the tape cartridge. The cartridge contains a small plastic door over the tape access area to protect the tape. The door is automatically opened when the tape cartridge is inserted in the Cartridge Tape Unit and closed when the cartridge is removed. The following precautions will help prolong the life of a tape cartridge and prevent data errors:

- Keep tape cartridges in a clean, dust-free area.
- Do not allow the light sensing windows (Fig. 16-4) to become smudged or dirty, as this may cause the tape to run off its spools. Under no circumstances should the windows be covered.

- Keep cartridges away from magnetic fields and ferromagnetic materials that might become magnetized. Strong magnetic fields can damage the magnetically recorded data on a tape.
- Use caution with cigarettes, cigars, and pipes around cartridges. Heat and contamination from a carelessly dropped ash can damage a tape.
- Do not expose cartridges to direct heat or strong sunlight. (Note **Environmental Specifications** in the section **Installation**.)
- Do not leave a tape cartridge inserted in the Cartridge Tape Unit drive for an extended period when the unit is not in use (such as overnight). This results in a temporary flat spot on the drive roller, causing the unit to be excessively noisy during the next few minutes of operation.
- Do not drop or throw a tape cartridge. Physical stress might bend the metal base plate or cause tape misalignment, resulting in data errors.
- Do not remove a tape cartridge from a Cartridge Tape Unit drive while the tape is in motion (while the BUSY light on the drive is lit). Inserting a tape cartridge in the drive automatically rewinds the tape. Do not remove a tape until rewinding is complete.

STORAGE DISPLAY MONITOR CONTROLS

The Storage Display Monitor has three exterior adjustment controls: FOCUS, WRITE THRU INTENSITY, and HARD COPY INTENSITY. Each control is easily adjusted with a screwdriver. The following information explains the location, purpose, and operation of each control.

FOCUS

The FOCUS control is on the rear panel of the Storage Display Monitor (Fig. 16-5) and adjusts the clarity of the information displayed on the screen.

The following example demonstrates the display variance as the FOCUS control is turned clockwise from its lowest extreme while characters are entered from the keyboard.

XX

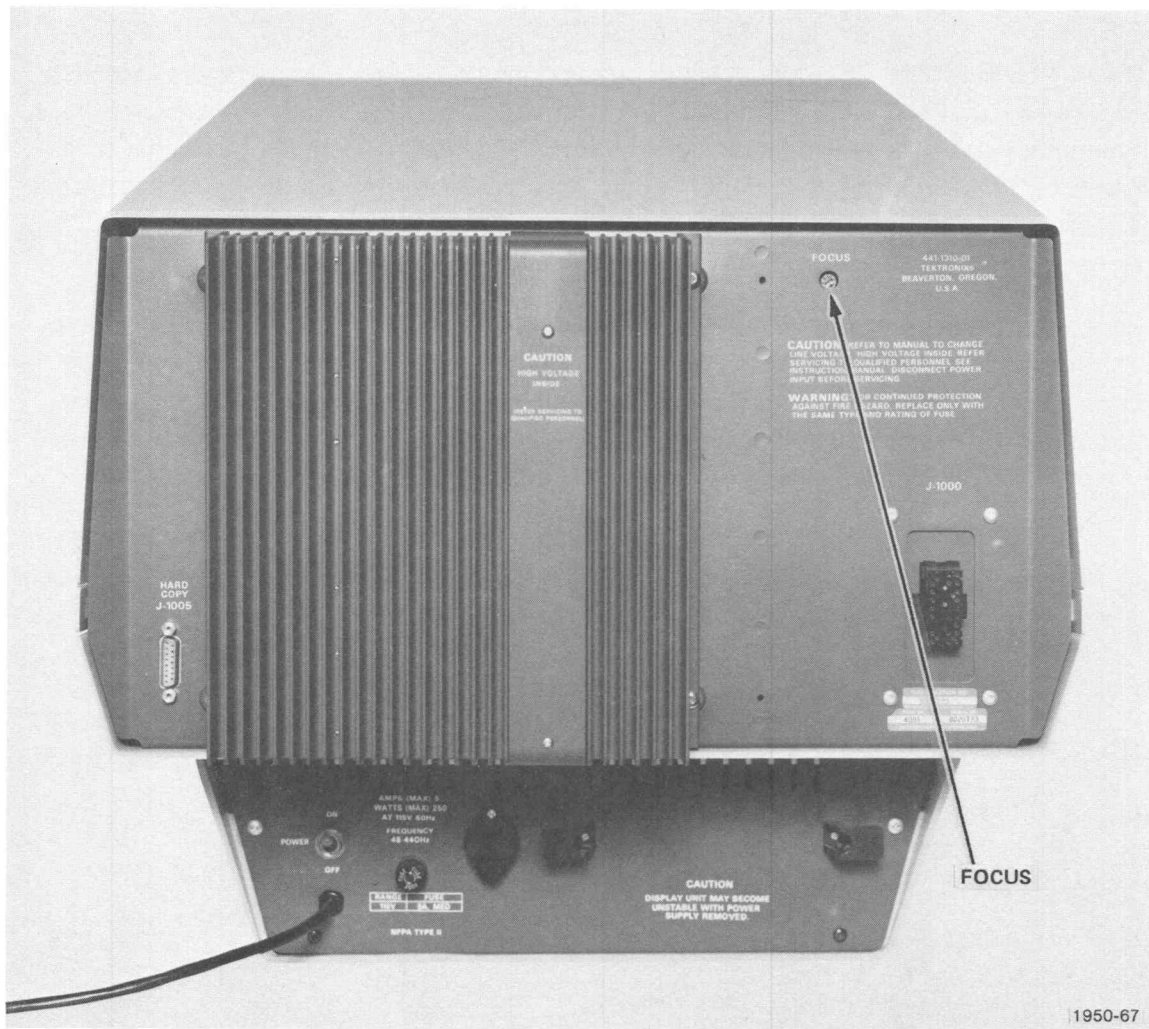


Fig. 16-5. FOCUS control.

WRITE THRU INTENSITY

The WRITE THRU INTENSITY control is on the right side of the Storage Display Monitor (Fig. 16-6) It controls the brightness of the information displayed on the screen in refresh. If the brightness is increased too much, the displayed information is stored on the screen. In other words, the image of the information remains visible on the screen until the screen is erased.

Fig. 16-7 shows the storage of an image displayed on the screen in refresh. To eliminate storage, turn the WRITE THRU INTENSITY control counterclockwise while moving the alphanumeric cursor with the space bar. After storage ceases (the previous image of the cursor does not remain visible when the cursor is moved to a new position on the screen), turn the control clockwise to the point of maximum brightness with no storage.

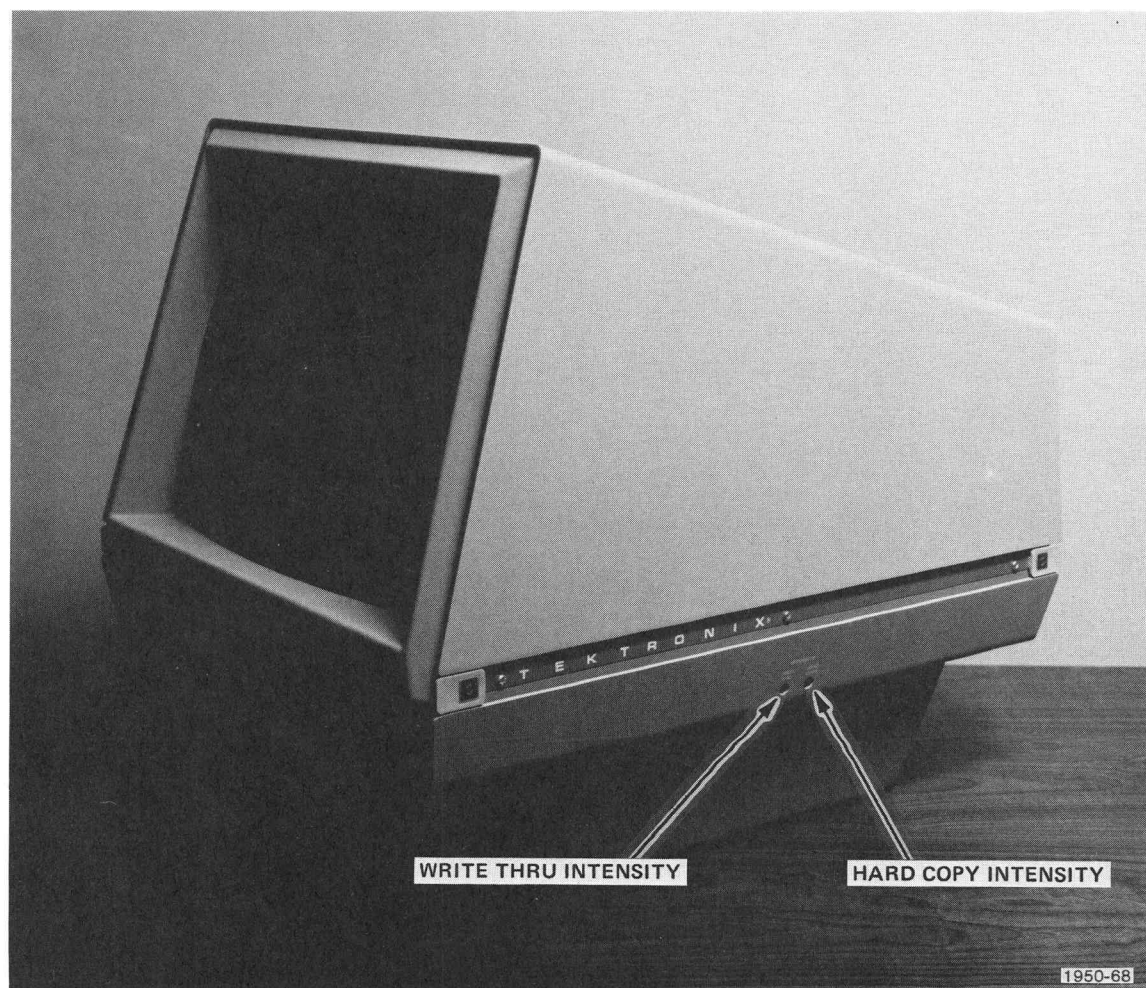


Fig. 16-6. WRITE THRU INTENSITY and HARD COPY INTENSITY controls.

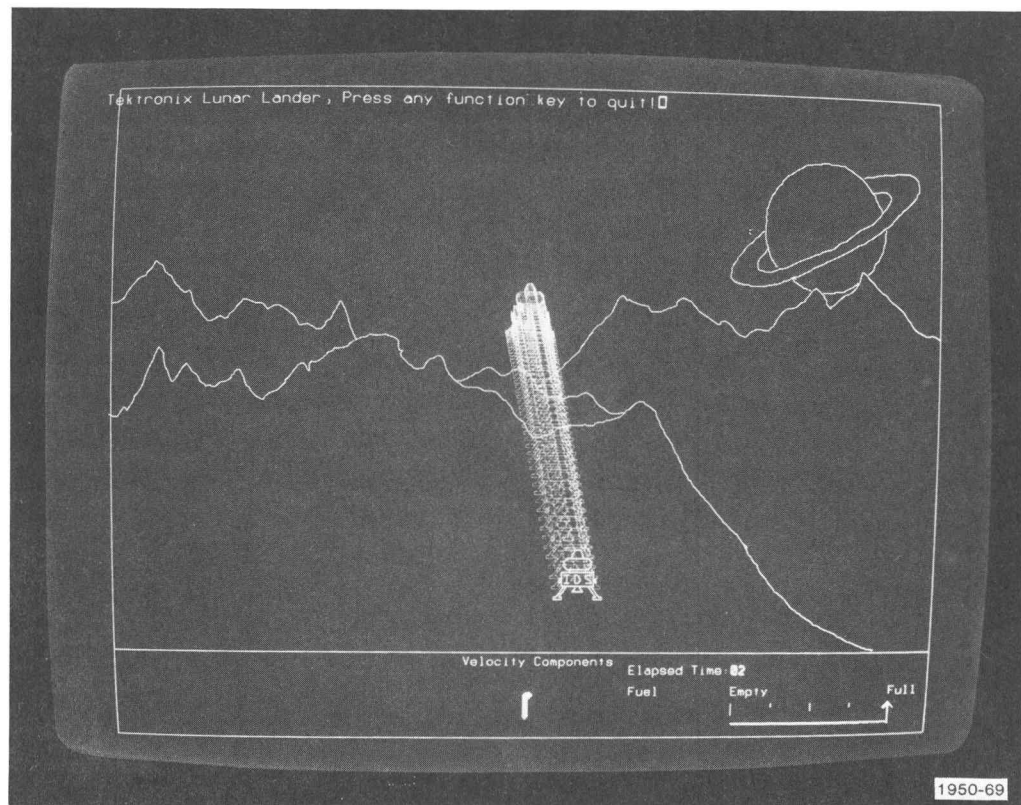
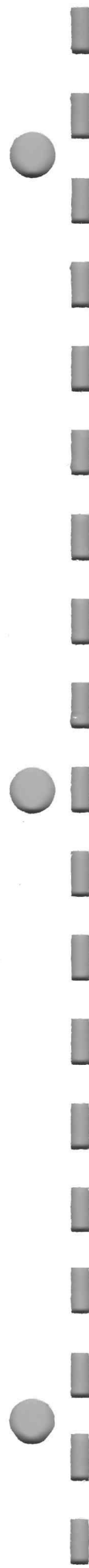


Fig. 16-7. Storage of a refresh image.

HARD COPY INTENSITY

The HARD COPY INTENSITY control is on the right side of the Storage Display Monitor (Fig. 16-6) and adjusts the contrast of a hard copy (if the user's 4081 includes a hard copy device). An optimal setting produces a hard copy without dark edges, without filling in of characters, and without storage on the screen from the hard copy scanning bar. Turning the HARD COPY INTENSITY control counterclockwise eliminates these effects, but turning it too far produces a copy that is incomplete and without detail.



APPENDIX A

GOS RESIDENT COMMAND SUMMARY

Appendix A

GOS RESIDENT COMMAND SUMMARY

Syntax	Description	Defaults
ASSIGN n=file or device	Assign a logical unit to a device or file.	file dev: =USR: .ext =none device dev: =KB:
CHARSIZE n	Set the Display Monitor character size.	none
CLOSE	Close open files, release dynamic memory (clear refresh graphics), reassign all logical units to KB:, release user-definable interval control blocks, and display Exit .	none
CONTINUE	Continue the execution of a suspended program.	none
LOAD file [@ hex loc] [arg]	Load a program into memory.	file dev: =USR: .ext =.OBJ hex loc @.GOSTOP
LU	Display a list of the 16 user-definable logical units and the device or file assigned to each one.	none

GOS RESIDENT COMMAND SUMMARY

Syntax	Description	Defaults
RUN file [@hex loc] [arg]	Load a program into memory and begin execution of the program.	file dev: =USR: .ext=.OBJ hex loc @.GOSTOP
START [@hex loc] [arg]	Start execution of a program loaded in memory.	hex loc @starting address of the last program loaded.
E hex loc	Display the contents of a memory location.	none
+	Display the contents of the next halfword after the last location examined.	none
-	Display the contents of the halfword before the last location examined.	none
D hex value	Deposit the specified hex value in last location examined.	none
*	Indicate a comment.	none

APPENDIX B

GOS UTILITY PROGRAM SUMMARY

Appendix B

GOS UTILITY PROGRAM SUMMARY

Syntax	Description	Defaults	Wild Cards	Switches
ATTRIB [:H] file [:W][:R] [:V] [,file ...]	Assign or remove read and/or write-protect file attributes.	file dev: =USR: .ext = none	allowed	;H display help message ;W write-protect attribute ;R read-protect attribute ;V verify
BATCH [:H] file [:L]	Execute commands listed in a batch file.	file dev: =USR: .ext = .BAT	not allowed	;L list each command when executed
COPY [:H] to file/device = from file/device[,file ...] [:V][:B]	Copy the data from a file or device to another file or device.	to/from file dev: =USR: .ext = none to/from device dev: none	* allowed ? only on right of "="	;H display help message ;V verify ;B binary graphics file
DELETE [:H] file[,file ...][:V]	Delete a file.	file dev: =USR: .ext = none	allowed	;H display help message ;V verify
DIR [:H] device/file[:B] [:E][:F[:n]][:L] [to device/file]	Display directory of file structure (or copy directory to device/file).	device dev: =USR: file dev: =USR: .ext = .* to device dev: =DC: to file dev: =USR: .ext = .LST	allowed	;H display help message ;B include bad blocks ;E include empty blocks ;F fast format ;f:n fast format, n char/line ;L do not list individual files
DISPLA [:H] file [,file]	Display graphics file.	file dev: =USR: .ext = .PLT	allowed	;H display help message

GOS UTILITY PROGRAM SUMMARY

Syntax	Description	Defaults	Wild Cards	Switches
FORMAT [:,H] device/file [[library size]] [:,N:name][:E:n] [:,X:n]	Format a file-structured device or library file.	device dev: = none file dev: = USR: .ext = .LIB	not allowed	;H display help message ;N:name specify the structure identifier ;E:n reserve n directory entries ;X:n reserve n extra bytes in each directory entry
HELP utility [,to device/file]	Display help message for GOS utility program.	utility dev: = assumed SYS: .ext = .HLP to device dev: = DC: to file dev: = USR: .ext = .LST	not allowed	none
PDBDMP [to file/device =] file	Display ASCII listing of Picture Data Base file (or copy listing to device/file).	file dev: = USR: .ext = .PDB to device dev: = DC: to file dev: = USR: .ext = .LST	not allowed	none
PLOT file	Draw graphics file on Plotter.	file dev: = USR: .ext = .PLT	not allowed	none
PRINT file	List ASCII file on Line Printer.	file dev: = USR: .ext = .LST	not allowed	none
RENAME [:,H] new file/name = old file/device[:,V] [,new ... = old ...]	Rename a file or structure identifier.	file dev: = USR: .ext = none new name name = none device dev: = none	* allowed ? only on right of "="	;H display help message ;V verify

Syntax	Description	Defaults	Wild Cards	Switches
SET [:,H] option [,option ...]	Set system parameters.	option none	not allowed	;H display help message
SQUISH [:,H] device/file[:,D]	Move empty space between files to the end of the file structure.	device dev:=none file dev:=USR: .ext=.LIB	not allowed	;H display help message ;D SQUISH only the directory
SYSTAT [to file/device]	Display current system status information (or copy to file/device).	to file dev:=USR: .ext=.LST to device dev:=DC:	not allowed	none
TYPE [:,H] file [,file ...]	Display ASCII file.	file dev:=USR: .ext=.ASM	allowed	;H display help message

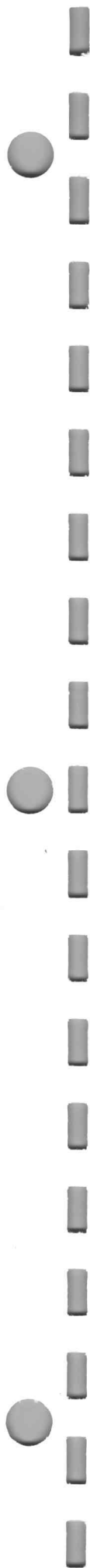
APPENDIX C

GENERAL SYSTEM MESSAGES

Appendix C

GENERAL SYSTEM MESSAGES

Message	Explanation
#	GOS command processor prompt. Enter a command or program to be executed.
\$	Argument prompt. An argument, usually a filename, is required and was not specified. Enter an argument or press the SHIFT-ESC keys to exit.
*	Argument prompt. A required argument was not specified. Enter an argument or press the SHIFT-ESC keys to exit.
Pause	Command or program execution is halted. Enter CON to continue execution or CLO to stop execution.
Exit	Command or program execution is finished.
< Attn >	The SHIFT-ESC keys were pressed and the 4081 is operating in Command mode.
< Del >	The ESC key was pressed and the current line is deleted.
< Host >	The SHIFT-BREAK keys were pressed and the 4081 is operating in Host mode.
< Local >	The SHIFT-RETURN keys were pressed and the 4081 is operating in Program mode (if a program is running on the 4081) or Command mode.



APPENDIX D

GOS STANDARD SYMBOLS

Appendix D

GOS STANDARD COMMAND AND PROGRAM SYNTAX SYMBOLS

Symbol	Description
file	A GOS standard file specifier. The GOS standard file specifier has the following format: device:filename.extension
device	A device mnemonic. The GOS device mnemonic consists of 2-3 characters followed by a colon(:).
program argument	A value, device, or file that is operated on by a program.
hex location	A specified location in memory. The address of the memory location is given in hexadecimal notation. Four hexadecimal digits are required to specify a 16 bit memory location (each hexadecimal digit represents four binary digits or bits). For the LOAD, RUN, and START commands, the hexadecimal location is preceded by the character @.
[]	Information enclosed in brackets is optional.
/	Library file identifier (e.g. the file SYSEQU.ASM in the library SYSLIB.LIB is specified as SYSLIB/SYSEQU.ASM).
:	1. Device identifier; follows the device mnemonic (e.g. FD0:). 2. Separates a switch from its value (e.g. ;N:NAME).
;	Switch identifier; precedes the switch (e.g. ;H).
.	Filename extension identifier; precedes the extension (e.g. .OBJ).
#	GOS command processor input request.

GOS STANDARD SYMBOLS

Symbol	Description
\$,*	General purpose input requests (input depends on the program running).

NOTE

In the examples of command and program syntax in this manual, the characters printed in boldface type represent the information that is actually displayed on the screen by GOS or the running program. The operator need type only the characters that are not printed in boldface type.

APPENDIX E

AVAILABLE 4081 DOCUMENTATION

Appendix E

AVAILABLE 4081 DOCUMENTATION

4081 Standard System

The following manuals are included as part of the standard 4081 Graphic System:

- 4081 Operator's Reference
- 4081 Operator's Guide
- Plot 80: 4014 Emulator
- Plot 80: IGT User's
- Plot 80: IGT Host Support User's
- Plot 80: IGT Host Support System's
- Plot 80: IGT Host System Programmer's Reference Guide
- Plot 80: Picture Data Base
- 4081 Verification Program
- Plot 80: Digitizer

4081 Operator's Reference

The 4081 Operator's Reference provides an introduction to the 4081 for all 4081 operators, programmers, and technicians. The manual contains a general description of the 4081 and its components, a list of the available options, a practice session for beginning 4081 operators, and descriptions of the demonstration programs, resident commands, and utility programs. The manual also includes sections on data communications in Host mode, files, error messages, installation, maintenance, and general operating information.

4081 Operator's Guide

The 4081 Operator's Guide provides a summary of most of the common 4081 operations described in the 4081 Operator's Reference.

Plot 80: 4014 Emulator

The Plot 80: 4014 Emulator manual describes how to run the Plot 80: 4014 Emulator program on the 4081 and how to operate the 4081 when the 4014 Emulator program is running. The manual also contains a list of differences between the 4014 Emulator program and the TEKTRONIX 4014-1 Computer Display Terminal.

Plot 80: IGT User's

The Plot 80: IGT User's manual provides information on the Plot 80: Interactive Graphic Terminal (IGT) package from the point of view of the 4081 programmer. The IGT commands are described in the form in which they are sent from the host computer to the 4081, in parameter blocks of two or more bytes. The manual also contains a detailed explanation of the IGT communications protocol.

Plot 80: IGT Host Support User's

The Plot 80: IGT Host Support User's manual provides information on the Plot 80: Interactive Graphics Terminal (IGT) package from the point of view of the host computer applications programmer. The manual contains descriptions of the IGT FORTRAN host support subroutines. The manual also includes sections on IGT communications protocol, graphics programming concepts, running the IGT program on the 4081, and implementing the IGT host support package on a host computer.

Plot 80: IGT Host Support System's

The Plot 80: IGT Host Support System's manual provides a host computer systems programmer with the information needed to modify the IGT FORTRAN host support subroutines. The manual also contains information on the user-written input/output and translation subroutines necessary to implement the IGT host support package on a host computer. A sample implementation on a host computer is included.

Plot 80: IGT Host System Programmer's Reference Guide

The Plot 80: IGT Host System Programmer's Reference Guide provides a summary of the IGT FORTRAN host support subroutines, brief definitions of the communications parameters, a list of the IGT function code parameter blocks sent by IGT to the host computer, a list of error codes, and an ASCII code chart.

Plot 80: Picture Data Base

The Plot 80: Picture Data Base manual describes the Plot 80: Picture Data Base, the data structure used by the Plot 80: IGT Package and the Plot 80: Digitizer program to store graphics information. Brief explanations of graphics programming concepts (for example, virtual space, setpoints, rotation, scaling, intensity, texture, long and short vectors, and instances) are included.

4081 Verification Program

The 4081 Verification Program manual describes how to run the 4081 Verification Program. The Verification Program is a sequence of programs that test the functioning of

the 4081 Graphic System's hardware components. The manual provides the operator with the information necessary to prepare the 4081 and individual devices for the tests and interpret the test results.

Plot 80: Digitizer

The Plot 80: Digitizer manual describes how to run the Plot 80: Digitizer program on the 4081 and how to create and store pictures using Digitizer. Digitizer can be run only on a 4081 Graphics System that includes one of the optional 4953/4954 Graphics Tablets. The manual includes a sample Digitizer session.

4080A01 Plot 80: Programming Support Package

The following manuals are included as part of the 4080A01 Plot 80: Programming Support Package:

- Plot 80: GOS Programmer's Reference
- Plot 80: Assembly Language Programming
- Plot 80: GOS Text Editor and Corrector (TECO)
- Plot 80: Library Linker/Loader
- Plot 80: RAID Debugging Aids
- Plot 80: GOS Pocket Reference

Plot 80: GOS Programmer's Reference

The Plot 80: GOS Programmer's Reference provides an assembly language programmer with the information needed to write and run programs on the 4081 under the control of the Graphic Operating System (GOS). The manual contains descriptions of the important GOS concepts (for example, device independence, memory allocation, file and device control, and graphics programming), special GOS memory locations and tables (for example, the Job Status Word, the Job Error Code, and the System Constants Table), and the GOS Supervisor Calls (SVC's).

Plot 80: Assembly Language Programming

The Plot 80: Assembly Language Programming manual provides an assembly language programmer with the information needed to write 4081 programs using the Plot 80: Assembly Language. The manual includes descriptions of the complete Plot 80: Assembly Language instruction set and pseudo-op statements (special directions to the Assembler). The manual also contains information on creating, assembling, and running a program, writing macroinstructions, floating-point operations, the Program Status Word (PSW), 4081 Processor interrupts, and input/output procedures.

Plot 80: GOS Text Editor and Corrector (TECO)

The Plot 80: GOS TECO manual describes how to create and edit program source files and alphanumeric text files using Plot 80: GOS TECO. The manual includes descriptions of all the TECO commands.

Plot 80: Library Linker/Loader

The Plot 80: Library Linker/Loader manual provides a programmer with the information needed to load, link, and execute both assembly language and FORTRAN programs. The manual includes descriptions of the complete set of Plot 80: Library Linker/Loader commands. The manual also contains information and examples on running the LINK program, linking an assembly language program with subroutines, linking a FORTRAN program with the routines in the Plot 80: FORTRAN IV Run Time Library, and using program overlays.

Plot 80: RAID Debugging Aids

The Plot 80: RAID Debugging Aids manual describes how to debug a program using the Plot 80: RAID Debugging Aids. The manual contains descriptions of all RAID directives. RAID's features are designed primarily for assembly language programs, but can be used to a limited extent for FORTRAN programs.

Plot 80: GOS Pocket Reference

The Plot 80: GOS Pocket Reference provides an assembly language programmer with a summary of the information needed to write, edit, and debug programs on the 4081. The booklet includes brief descriptions of the Plot 80: Assembly Language instruction set and pseudo-ops, GOS Supervisor Calls (SVC's), GOS TECO commands, Library Linker/Loader commands, and RAID directives.

4080A02 Plot 80: FORTRAN IV Compiler

The following manuals are included as part of the 4080A02 Plot 80: FORTRAN IV Compiler:

- Plot 80: FORTRAN IV User's
- Plot 80: FORTRAN IV Reference
- Plot 80: FORTRAN IV Run Time Library

Plot 80: FORTRAN IV User's

The Plot 80: FORTRAN IV User's manual provides a programmer with the information needed to create, compile, and execute Plot 80: FORTRAN IV programs on the 4081. The manual contains descriptions of the Plot 80: FORTRAN IV Compiler control statements

and Compiler error messages. The manual also includes information on writing assembly language subroutines that can be called by a FORTRAN program and listings of the assembly language code equivalents for the standard FORTRAN statements.

Plot 80: FORTRAN IV Reference

The Plot 80: FORTRAN IV Reference manual is a standard FORTRAN IV programmer's reference. The manual contains descriptions of the 1966 ANSI Standard FORTRAN IV statements, extensions, and intrinsic and external functions.

Plot 80: FORTRAN IV Run Time Library

The Plot 80: FORTRAN IV Run Time Library manual describes the set of FORTRAN-related and GOS-related routines that are contained in the Plot 80: FORTRAN IV Run Time Library. The manual includes a brief introduction to graphics programming in FORTRAN.

4081 Service

The following 4081 Graphic System service manuals must be ordered separately:

- 4081 Technical Data
- 4081 Schematics
- 4081 System Parts

4081 Technical Data

The 4081 Technical Data manual provides the service procedures and theory of operation needed to install, maintain, troubleshoot, and repair the 4081 Graphics System. The manual contains information on the specifications, maintenance, and circuitry for the following 4081 components and options:

- Storage Display Monitor
- Display Controller
- Alphanumeric Keyboard and Controller
- RS-232 Communications Interface
- Cartridge Tape Unit
- MOS Memory
- Processor
- Power Supply and Backplane
- Graphics Tablet and Controller (4953/4954, Option 30)

The manual is intended primarily for qualified technicians. However, some of the programming information for the individual components (for example, the status and command bytes for each device) may be of interest to 4081 programmers.

4081 Schematics

The 4081 Schematics manual contains the schematics of all standard circuitry in the 4081 Graphic System. In addition, schematics are included for the following options:

- Expansion Package (Option 31)
- Graphics Tablet and Controller (4953/4954, Option 30)

The manual also contains block diagrams, connector sheets, and component location illustrations. The manual is intended for use with the 4081 Technical Data and 4081 System Parts manuals.

4081 System Parts

The 4081 System Parts manual lists the mechanical and electrical parts used in the standard 4081 Graphic System. The listings contain the locations, numbers, descriptions, and Tektronix part numbers of replaceable parts. In addition, parts listings are included for the following options:

- Additional 32K Bytes of Memory (Option 22)
- Expansion Package (Option 31)
- Additional Cartridge Tape Unit Drive (Option 33)
- Graphics Tablet (4953/4954, Option 30)

Optional Devices

The documentation available for each of the optional devices that can be connected to the 4081 Graphic System is listed beneath the individual device. The operator's (or user's) manual is included with the device. The service manual for each device must be ordered separately.

Generally, the operator's (or user's) manual contains information on installing, operating, and maintaining the device. The operator's manual also usually includes some programming information (for example, the status and command bytes for the device).

The service manual provides the service procedures and theory of operation needed to install, maintain, troubleshoot, and repair the device. The manual includes device schematics and a list of replaceable parts. The service manual is intended primarily for qualified technicians.

The 4905 Mass Storage Module

Operator's

- 4905 Mass Storage Module Operator's

Service

- 4905 Mass Storage Module Service
- 119-0845-00 Flexible Disc Drive
- 119-0852-00 Hard Disc Drive
- 119-0851-00 Selector Channel
- 119-0853-00 Hard Disc Controller

The 4631 Hard Copy Unit

Operator's

- 4631 Hard Copy Unit User's

Service

- 4631 Hard Copy Unit Service

The 4641 Character Printer

Operator's

- The 4641/4641-1 Character Printer Operator's

Service

- The 4641/4641-1 Character Printer Service

The 4662 Interactive Digital Plotter

Operator's

- 4662 Interactive Digital Plotter User's

Service

- 4662 Interactive Digital Plotter Service

The 4953/4954 Graphics Tablets

Operator's

- 4953/4954 Option 30 Graphics Tablet Operator's

Service

See the 4081 Technical Data manual for service information, the 4081 Schematics manual for Graphics Tablet schematics, and the 4081 System Parts manual for a list of replaceable parts.

The Hexadecimal Display Panel

Operator's and Service

- 067-0772-00 Hexadecimal Display Panel

The PROM Bootstrap Loader (Romulan Loader)

Operator's and Service

- 067-0794-00 Romulan Loader

APPENDIX F

ASCII CODE CHART

Appendix F

ASCII CODE CHART

LSD (HEX)	MSD (HEX)				0	1	2	3	4	5	6	7
	BITS				0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
	B4	B5	B6	B7	CONTROL		HIGH X & Y GRAPHIC INPUT		LOW X		LOW Y	
0	0	0	0	0	NUL	DLE	SP	0	@	P	\	p
					0 (0)	10 (16)	20 (32)	30 (48)	40 (64)	50 (80)	60 (96)	70 (112)
1	0	0	0	1	SOH	DC1	!	1	A	Q	a	q
					1 (1)	11 (17)	21 (33)	31 (49)	41 (65)	51 (81)	61 (97)	71 (113)
2	0	0	1	0	STX	DC2	"	2	B	R	b	r
					2 (2)	12 (18)	22 (34)	32 (50)	42 (66)	52 (82)	62 (98)	72 (114)
3	0	0	1	1	ETX	DC3	#	3	C	S	c	s
					3 (3)	13 (19)	23 (35)	33 (51)	43 (67)	53 (83)	63 (99)	73 (115)
4	0	1	0	0	EOT	DC4	\$	4	D	T	d	t
					4 (4)	14 (20)	24 (36)	34 (52)	44 (68)	54 (84)	64 (100)	74 (116)
5	0	1	0	1	ENQ	NAK	%	5	E	U	e	u
					5 (5)	15 (21)	25 (37)	35 (53)	45 (69)	55 (85)	65 (101)	75 (117)
6	0	1	1	0	ACK	SYN	&	6	F	V	f	v
					6 (6)	16 (22)	26 (38)	36 (54)	46 (70)	56 (86)	66 (102)	76 (118)
7	0	1	1	1	BEL	ETB	/	7	G	W	g	w
					7 (7)	17 (23)	27 (39)	37 (55)	47 (71)	57 (87)	67 (103)	77 (119)
8	1	0	0	0	BS	CAN	(8	H	X	h	x
					8 (8)	18 (24)	28 (40)	38 (56)	48 (72)	58 (88)	68 (104)	78 (120)
9	1	0	0	1	HT	EM)	9	I	Y	i	y
					9 (9)	19 (25)	29 (41)	39 (57)	49 (73)	59 (89)	69 (105)	79 (121)
A	1	0	1	0	LF	SUB	*	:	J	Z	j	z
					A (10)	1A (26)	2A (42)	3A (58)	4A (74)	5A (90)	6A (106)	7A (122)
B	1	0	1	1	VT	ESC	+	;	K	[k	{
					B (11)	1B (27)	2B (43)	3B (59)	4B (75)	5B (91)	6B (107)	7B (123)
C	1	1	0	0	FF	FS	,	<	L	\	l	!
					C (12)	1C (28)	2C (44)	3C (60)	4C (76)	5C (92)	6C (108)	7C (124)
D	1	1	0	1	CR	GS	-	=	M]	m	}
					D (13)	1D (29)	2D (45)	3D (61)	4D (77)	5D (93)	6D (109)	7D (125)
E	1	1	1	0	SO	RS	.	>	N	^	n	~
					E (14)	1E (30)	2E (46)	3E (62)	4E (78)	5E (94)	6E (110)	7E (126)
F	1	1	1	1	SI	US	/	?	O	_	o	RUBOUT (DEL)
					F (15)	1F (31)	2F (47)	3F (63)	4F (79)	5F (95)	6F (111)	7F (127)

APPENDIX G

GLOSSARY

Appendix G

GLOSSARY

alphanumeric cursor	The small rectangle (or underline character) that is displayed on the 4081's Display Monitor to indicate the current position of the writing beam. Characters typed on the keyboard are displayed starting at the current position of the alphanumeric cursor.
argument	A value, device, or file that is operated on by a program or resident command (also called a parameter). The term also refers to the part of the program or command syntax that specifies the value, device, or file.
ASCII Code	A standardized binary code for representing alphanumeric characters, symbols, and special "control" characters. "ASCII" is an acronym for American Standard Code for Information Interchange.
ASCII file	A file that contains alphanumeric characters, symbols, and special "control" characters stored in binary ASCII code. Generally, most ASCII files contain either text information or program source code.
assembler	A program that converts an assembly language program source file into a program load file that can be executed by a computer.
bad block	A defective area on a mass storage medium. Under GOS, data cannot be stored on or retrieved from a bad block.
binary	The base 2 number system. There are two binary digits: 0 and 1.
bit	An abbreviation of binary digit. A bit is the basic unit of information storage. A bit, at any one time, has a value of either 0 or 1.

GLOSSARY

block	A set of data bytes in memory or on a formatted mass storage medium which are handled as a unit, particularly in reference to input/output operations. Under GOS, the standard block size is 256 bytes.
byte	A group of eight consecutive binary digits (or bits).
Command mode	A 4081 operating mode. When the 4081 is operating in command mode, characters typed on the keyboard are sent to the GOS command processor. The 4081 must be in Command mode in order to run a program or execute a resident command from the keyboard.
command processor	The section of the GOS program that interprets a line of keyboard input as a program to be run or a resident command to be executed.
compiler	A program that converts a high-level language program source file (for example, a FORTRAN program source file) into a program load file that can be executed by a computer.
Concise Command Language(CCL)	A way of running a program under GOS by typing the program name and its argument on one line of keyboard input.
crosshair cursor	The default graphic input cursor. The crosshair cursor consists of a vertical line and a horizontal line that bisect each other.
debug	To detect, locate, and correct errors in a computer program.
default	A predetermined value that is assumed if no other value is specified.
device drive	The part of a device in which the medium (for example, a flexible disc) is inserted. The drive contains the mechanics and circuitry that allow data to be read from and written to the medium.
digitize	To convert points on a two-dimensional surface into digital X,Y coordinates that can be processed by a computer.

directory	The area reserved at the beginning of each file structure that contains a list of the files in the file structure and the information needed to locate each file.
edit	To modify the contents of a file.
execute	To initiate the processing of a sequence of instructions stored in memory.
extension	An optional 1-3 character addition to a filename which further identifies the contents of the file.
file	One or more blocks of data on a formatted mass storage medium (a tape cartridge, flexible disc, or hard disc) which is assigned a file specifier and accessed using the file specifier.
file specifier	A means to name and access a file. The file specifier supplies the information needed to locate a file for processing. In addition, the file specifier allows the operator to identify the contents of a file.
file structure	A formatted mass storage medium or a formatted portion of a formatted medium that has the capacity to contain a number of files. Each file structure includes a directory which lists the files contained in the file structure.
file-structured device	A mass storage device connected to the 4081 that allows a 4081 operator to first format a mass storage medium and then to create and access files on the formatted medium.
format	To prepare a mass storage medium for the storage of files.
.GOSTOP	A GOS term whose value is the address of the first halfword after the section of memory in which the GOS program is loaded. It is the lowest possible memory location available for loading a user program.

GLOSSARY

graphic input (GIN)

The transmission of X,Y coordinates from a graphic input device (the Joystick, Graphics Tablet, or Plotter) to a running program. A graphic input device converts points on a two-dimensional surface into digital X,Y coordinates that can be processed by the 4081.

graphic input cursor

A picture (generally, a crosshair cursor) that is displayed on the 4081's Display Monitor during graphic input operations. The graphic input cursor allows the operator to locate and select points on the screen. The cursor is moved across the screen by means of the graphic input device (either the Joystick, the Graphics Tablet pen, or the Plotter pen).

Graphic Operating System (GOS)

The operating system for the 4081. GOS consists of a group of software routines that supervise the use and operation of the 4081 Graphic System. GOS performs such basic tasks as input/output processing, interrupt handling, and file and device control. In addition, GOS is designed to simplify the writing and implementation of graphics application programs.

graphics file

A file that contains graphics information in the form of screen X,Y coordinates. Generally, graphics files are created by storing the output of SVC 15 or FORTRAN graphics instructions in a file.

halfword

A group of 16 consecutive binary digits (or bits); two bytes.

hard copy

A printed copy of the information stored in the 4081's memory or displayed on the 4081's Display Monitor.

hard copy scanning bar

The bar of light that sweeps across the Display Monitor screen when a hard copy is produced using the TEKTRONIX 4631 Hard Copy Unit.

hardware

A general term that refers to the physical components of a computer system.

hexadecimal notation	The representation of a value using the base 16 number system. There are 16 digits: 0 to 9 and A to F. Generally, hexadecimal notation is used to represent binary numbers with each hexadecimal digit corresponding to four consecutive binary digits (bits).
Hold state	A state of the 4081's Display Monitor in which the screen becomes darker and the intensity of the screen image is dimmed. Hold state takes effect if no information is output to the Display Monitor for approximately 90 seconds.
host computer	The computer system with which the 4081 operator communicates when using the 4081 as a terminal for data communications.
Host mode	A 4081 operating mode. When the 4081 is operating in Host mode, characters typed on the keyboard are sent through the primary communications interface. Generally, the primary communications interface is connected to a host computer, but it also can be connected to an external device (for example, a plotter or line printer).
input	To transfer information from an external device to the internal storage (the memory) of a computer. (Same as read .) The term also can refer to the transferred information.
library file	A formatted portion of a device that contains a directory and a collection of the files listed in the directory. A library is specified by a filename (with a standard extension of .LIB).
link	To connect two or more separately assembled or compiled programs or routines into one single program or system of programs.
load	To transfer a program load file from an external device into memory in order to execute the program.
logical unit (LU)	A number (from 0-15) assigned to a physical device connected to the 4081. Under GOS, all input/output operations are usually performed to and from logical units, not the devices directly. A device can be assigned to more than one logical unit.

GLOSSARY

mass storage medium	The material on which digital information is recorded for long-term storage. For example, flexible discs, hard discs, and tape cartridges are mass storage media.
modem	A device that converts a digital signal from a computer or terminal into an analog signal that can be transmitted over telephone lines. Modems generally are used for long-distance data communications between computers or between a computer and a terminal. "Modem" is an acronym for MOdulator/DEModulator.
monitor viewport (MVP)	The area of the 4081's Display Monitor screen on which keyboard input and GOS messages are displayed.
nested file	A file that is created within a library file. A nested file is listed in the library file's directory.
output	To transfer information from the internal storage (the memory) of a computer to an external device. (Same as write .) The term also can refer to the transferred information.
overlay	A module (or section) of a program that is loaded into memory as it is needed during the execution of a program.
Picture Data Base (PDB)	A format for the storage of binary graphics information. Picture Data Base files are created by the Plot 80: software packages Interactive Graphics Terminal (IGT) and Digitizer.
primary directory	The directory of a formatted mass storage medium. The primary directory is located at the beginning of the medium.
processor	An electronic device that is capable of receiving instructions and data, operating on the information, then transmitting the results.
program	A sequence of instructions which directs a computer to perform a specific task.

program load file	A file that contains a program or section of a program. The program load file can be loaded into the 4081 memory and executed.
Program mode	A 4081 operating mode. When the 4081 is operating in Program mode, the program running on the 4081 determines the destination of characters typed on the keyboard. The 4081 enters Program mode when a program is run and remains in Program mode until the program pauses, exits, or is terminated by GOS because of an error.
program source file	A file that contains the assembly language or FORTRAN coding for a program or section of a program. The program source file must be converted to a program load file by an assembler or compiler before it can be run.
Program Status Word (PSW)	A dedicated storage area in the 4081's central processing unit (CPU) that defines the state of the CPU at a specific time. The PSW contains 32 bits. Bits 0-7 control interrupts. Bits 8-11 are always clear (0). Bits 12-15 are called the Condition Code and reflect the result of a previous instruction. Bits 16-31 contain the location counter that represents the location in memory of the instruction currently being executed.
prompt characters	An alphanumeric character that is displayed on the 4081's Display Monitor by GOS or a running program to request an operator response. The response, generally typed on the keyboard, depends on the prompt character displayed and the program running on the 4081.
prompt light	One of the red lights located on the upper right of the 4081's Keyboard Unit. The prompt lights generally indicate the status of the keyboard and the monitor viewport (MVP). The prompt lights are programmable, however, and their meaning can vary from program to program.
read	To transfer information from an external device to the internal storage (the memory) of a computer. (Same as input .)

GLOSSARY

read-protect	To prevent the information stored on a mass storage medium from being transferred to the internal storage (the memory) of a computer.
refresh	A method of displaying computer graphics on a display monitor by continually drawing and redrawing a picture on the screen, generally at a rate of 30 to 60 times a second.
resident command	A command routine stored on the GOS IPL tape cartridge which is loaded into the 4081 memory as part of the GOS program.
run	To load a program load file from a file-structured device into memory and then execute the program.
software	A general term that refers to the programs designed to simplify the operation and programming of a computer system. The term also can refer to any computer program or group of programs.
storage	A method of displaying computer graphics on a display monitor by drawing a picture once on the screen. The picture is "stored" on the screen. In other words, it remains visible until the screen is erased.
structure identifier	The name (of 1-6 characters) assigned to a file structure when the structure is formatted or renamed. A structure identifier is intended as an operator's aid for further identifying a file structure. It is ignored by GOS and is not part of the standard file specifier.
switch	Under GOS, a letter preceded by a ; (semi-colon) which indicates a specific program option. When running a program, the switches are typed on the same line of keyboard input as the program filename.
system device	A term that refers to the file-structured device drive in which the system disc (or tape cartridge) is inserted. The system disc (or tape) contains the GOS utility programs.

TTY character set	The standard ASCII character set of Teletype, Inc. It consists of the 64 characters with ASCII codes 32(SP) to 95(—), inclusive.
user device	A term that refers to the file-structured device drive in which the mass storage medium containing the 4081 user's files is inserted.
utility program	A GOS program stored in a file on the system disc (or tape) which is provided to ease some of the routine tasks of a 4081 operator.
View state	A state of the 4081's Display Monitor in which the intensity of the screen and the image displayed on the screen is set for normal viewing.
wild card	<p>A character used in a file specifier to represent any or all characters. There are two wild card characters:</p> <p>? represents any character in a specific position. * represents any set of characters in a specified field (filename or extension).</p>
write	To transfer information from the internal storage (the memory) of a computer to an external device. (Same as output .)
write-protect	To prevent the transfer of information from the memory of a computer to a mass storage medium.

INDEX

INDEX

alphanumeric cursor	7-7, 11-46, 11-47, G-1
ASCII code	3-5, 7-16, 12-4 F-1, G-1
ASCII code chart	F-1
ASCII files	8-32, 8-33, 8-34 11-39, 11-55, G-1
Assembler	4-19, 8-32, E-3
ASSIGN	9-5, 12-16, A-1
asynchronous communications	12-7
ATTRIB	11-8, B-1
bad blocks	8-43, 11-21, G-1
BATCH	5-6, 7-6, 11-10, B-1
baud rate	11-53, 12-5, 12-13
BREAK	7-18, 12-6, 12-15
break signal	11-53, 12-6, 12-10, 12-15
Cartridge Tape Unit	3-7, 4-7, 4-17 7-5, 8-3, 8-30
BUSY light	7-5, 8-6
cleaning	15-3
data transfer rate	8-4
formatting a tape cartridge	8-30, 11-29
inserting a tape cartridge	8-6
tape cartridge capacity	8-4
tape cartridge care	16-7
tape cartridge respooling	15-14
write-protecting	5-5, 8-7
central processing unit (CPU)	2-7, 3-3
CHARSIZE	5-20, 9-7, A-1
circuit breaker	7-4, 16-3
CLOSE	9-8, 12-16, A-1
Command mode	7-11, 7-26, G-2
command processor	7-11, 10-7, 13-3, G-2
COMMENT(*)	5-19, 9-22, A-2
communications interface	3-8, 12-9, 12-10
Concise Command Language (CCL)	10-7, G-2

INDEX

CONTINUE.....	5-18, 9-9, A-1
cooling vents.....	15-7
COPY.....	11-14, B-1
crosshair cursor.....	6-12, 11-45, 11-46, G-2
CTRL.....	7-19
CTRL-N.....	7-19, 7-21
CTRL-O.....	7-15, 7-20, 11-26, 11-55
CTRL-R.....	7-8, 7-19, 7-21
CTRL-S.....	7-15, 7-21, 11-26, 11-55
CUBE.....	3-13, 6-10
data communications.....	3-11, 7-13, 11-52, 12-3
DELETE.....	11-19, B-1
demonstration programs.....	3-13, 6-3
DEPOSIT (D).....	9-20, A-2
device independence.....	2-9, 8-58
device mnemonics.....	5-9, 8-31, 8-38, 11-52
Digitizer.....	3-12, 4-14, E-3
DIR.....	5-15, 11-21, B-1
directory.....	8-44, 8-52, 11-21 11-28, 11-49, G-3
directory header.....	8-46, 8-53
DISPLA.....	11-25, B-1
display processor.....	2-7, 3-6
echo.....	11-53, 12-6, 12-15
empty files.....	8-59
ESC (Escape).....	5-22, 7-23
EXAMINE (E).....	9-17, A-2
EXAMINE NEXT HALFWORD (+).....	9-18, A-2
EXAMINE PREVIOUS HALFWORD (-).....	9-19, A-2
Expansion Package.....	4-17
file creation.....	8-59
file extensions.....	8-32, G-3
file sizing.....	8-61, 11-15, 11-23, 11-35, 11-51
file specifiers.....	8-30, G-3
file-structured device.....	8-3, 8-31, G-3

Flexible Disc Unit	4-7, 8-30
data transfer rate	8-11
flexible disc capacity	8-10
formatting a flexible disc	8-30
inserting a flexible disc	8-11
write-protecting	5-8, 8-13
FORMAT	8-3, 8-39, 8-47, 11-27, B-2
FORTTRAN IV	8-33, E-4
Compiler	4-21
Run Time Library	4-21, E-5
4014 Emulator	3-10, E-1
4905 Mass Storage Module (also see Flexible Disc Unit and Hard Disc Unit)	4-7, 5-3, 14-5, 14-5
full duplex	12-7
function keys	3-6, 7-14, 7-16
fuse replacement	15-8
.GOSTOP	9-10, 9-14, G-3
graphic input (GIN)	2-5, 3-7, 4-11, 4-14, 11-43, G-4
graphic input cursor	6-12, 7-17, 11-45, G-4
Graphic Operating System (GOS)	3-5, 3-9, 5-4, E-3, G-4
graphics files	8-34, 11-14, 11-25, 11-37, G-4
Graphics Tablet	4-12, 11-47, E-8
HAND	3-13, 6-12
hard copy scanning bar	7-10, 16-11, G-4
Hard Copy Unit	4-7, 5-3, E-7
Hard Disc Unit	4-7, 8-30
data transfer rate	8-18
formatting a hard disc	8-30
hard disc capacity	8-18
inserting a hard disc cartridge	8-20
removing a hard disc cartridge	8-28
write-protecting	8-27
HELP	5-17, 11-33, B-2
Hexadecimal Display Panel	4-14, E-8
Hold state	7-10, 7-25, G-5
host computer communications (see data communications)	
Host mode	7-13, 7-25, 12-3, G-5

INDEX

Initial Program Load (IPL)	3-7, 4-16, 5-5, 7-5, 15-10
Interactive Graphics Terminal (IGT)	2-8, 3-10, E-2
Job Status Word (JSW)	7-9, 12-8
Joystick	2-7, 3-7, 7-14, 7-17
Keyboard Unit	3-6, 7-16
LANDER	3-13, 6-8
LF (Line Feed)	7-11, 7-12, 7-23
library files	8-35, 8-46, 8-50, 8-53 11-17, 11-20, 11-28, G-5
Library Linker/Loader	4-20, 4-22, E-4
LIFE	3-13, 6-17
LOAD	9-10, 9-15, 10-5, A-1
logical units (LU)	8-58, 9-5, 9-12, 13-3, G-5
LOGO	3-13, 6-4
LU	9-12, A-1
Memory	3-5, 4-17
modem	12-3, 12-4, G-6
monitor viewport (MVP)	7-7, 11-44, G-6
MOTOR	3-13, 6-14
MVP FULL condition	7-8, 7-19, 7-21, 7-24
nested files	8-35, 8-54, G-6
Null device	8-39, 11-53
overlays	4-20, 8-34, G-6
parity	11-53, 12-4, 12-12
PDBDMP	11-35, B-2
permanent files	8-59
Picture Data Base (PDB)	8-34, 11-25, 11-35, 11-37, E-2, G-6
PLOT	4-10, 11-37, B-2
Plotter	4-10, 11-37, E-7
power bus	7-4, 16-3
power cord	14-10, 14-11
power outlet box	14-10
power supply	14-9
POWER switch	7-4
primary communications interface	12-9, 12-11
PRINT	4-8, 11-39, B-2
Printer	4-8, 11-39, E-7
Program mode	7-12, 7-27, 12-8, G-7
Program Status Word (PSW)	4-14, 13-7, G-7
Programming Support Package	4-19, E-3

PROM Bootstrap Loader	4-16, E-8
prompt characters	5-5, 7-8, C-1, G-7
prompt lights	3-7, 7-14, 7-15, 9-20, 15-9, G-7
DELETE OUTPUT	7-15, 7-20
MVP FULL	7-8, 7-15
REQUEST INPUT	5-5, 7-15
STOP OUTPUT	7-15, 7-21
RAID	4-20, E-4
refresh graphics	2-5, 2-6, 3-6, 7-22, G-8
RENAME	11-40, B-2
RESET/PAGE	5-19, 7-7, 7-8, 7-24
resident commands	5-19, 9-3, G-8
RETURN	7-11, 7-12, 7-24
Romulan Loader (see PROM Bootstrap Loader)	
RS-232 interface	12-7
RUBOUT	5-21, 7-24
RUN	5-13, 9-14, 10-5, A-2
SET	5-12, 11-42, 12-10, B-3
SHIFT	7-25
SHIFT-BREAK	7-13, 7-25, 12-9
SHIFT-ESC	5-18, 7-11, 7-25
SHIFT-RESET	7-26
SHIFT-RETURN	7-13, 7-27
speeds	11-53, 12-5, 12-14
SQUISH	11-49, B-3
START	9-11, 9-15, 10-5, A-2
start bit	12-4
stop bits	11-53, 12-4, 12-12
Storage Display Monitor	3-6, 7-9, 15-10, 16-8
FOCUS control	16-8
HARD COPY INTENSITY control	16-11
MONITOR ERASE button	7-10
POWER switch	7-3
STORAGE HARD COPY button	7-10
VIEW button	7-10
WRITE THRU INTENSITY control	16-10
storage graphics	2-5, 2-6, 3-6, 4-7, G-8
structure identifier	8-40, 11-27, 11-40, G-8

INDEX

switches	11-6, G-8
SYSTAT.....	5-12, 11-51, 12-17, B-3
system device (SYS:)	5-10, 5-13, 8-17, 8-29 11-3, 11-43, 11-52, G-8
TAB.....	7-27
TECO.....	4-20, 7-23, 11-11, E-4
tentative files.....	8-59
TTY LOCK	5-22, 7-27
TYPE	11-55, B-3
type-ahead.....	7-8, 12-7
user device (USR:).....	5-13, 11-43, 11-52, G-9
utility programs.....	3-9, 11-3, 11-33, G-9
Verification Program	3-11, 15-8, E-2
View state.....	7-10, 7-25, G-9
wild cards.....	10-8, G-9
XLOGO	2-7, 3-13, 6-6

